MASTER EN SCIENCES INFORMATIQUES

Open source
guide à l'usage des administrations publiques

Turine, Vincent

Award date:
2006

Facultés Universitaires Notre-Dame de la Paix, Namur
Institut d'Informatique
Année académique 2005-2006

Open Source
Guide à l'usage des Administrations Publiques

Vincent Turine

Mémoire présenté en vue de l'obtention du grade de Maître en Informatique

Résumé

Le but de ce mémoire est de proposer aux administrations un outil d'aide à la décision concernant le passage à l'Open Source. En effet, il a été constaté que ce thème, bien qu'étant d'actualité, manquait de clarté et de visibilité pour les administrations wallonnes. Nous allons donc essayer de proposer un éclairage simple et concis de tout ce qu'il faut savoir à ce sujet.

Ce mémoire va donc proposer :

- un recueil d'informations pertinentes sur l'Open Source,
- un exposé des forces et faiblesses de l'Open Source à l'heure actuelle,
- et un guide qui devrait aider les administrations intéressées à réaliser une migration, dans de bonnes conditions, vers l'Open Source.

Mots-clés :

Open Source, logiciel libre, licence de logiciel, format de fichiers, guide de migration

Abstract

The purpose of this study is to propose a tool for decision when adopting Open Source. It must be stated that this topic despite its actuality suffers a lack of understanding and visibility from the Walloon public services. We shall therefore attempt to propose a simple and concise analysis on the major aspects of this topic.

This study will consist of :

- relevant information on Open Source,
- strengths and weaknesses of Open Source today,
- guidelines to help interested administrations to migrate to Open Source in optimal conditions.

Keywords :

Open Source, Free Software, Free License, file format, migration guidelines

Remerciements

Mes remerciements s'adressent en premier lieu à ma promotrice, Madame C. Lobet-Maris, pour son aide, sa disponibilité et ses conseils avisés.

Je tiens également à remercier Monsieur B. Genin et le personnel d'EASI-WAL qui m'ont aimablement accueilli pendant deux mois.

Je remercie aussi Mesdames M.-J. Baeken et A. Misonne et Messieurs L. Colson, Y. Cool, J.-F. Coutelier, D. Delaunois, J. Herman, O. Schneider et J.-D. Veuve qui ont acceptés de m'éclairer sur certains points.

Enfin et surtout, je remercie ma famille du soutien et de l'aide qu'elle m'a apportés tout au long de ces années d'études.

Table des Matières

Résumé....i
Abstract....i
Remerciements....iii
Liste des acronymes....1
Introduction....3
Chapitre 1 : L'Open Source se découvre....7
1 Définitions....7
1.1 Logiciel Libre (« Free Software »)....8
1.2 Open Source....8
1.3 Logiciel Libre vs Open Source....9
1.4 Philosophie....10
1.5 Utilisation des logiciels open source....11
2 Histoire de l'open source....11
2.1 L'Histoire....11
2.1.1 La première période : la préhistoire de l'informatique....12
2.1.2 La deuxième période : la naissance de l'open source....12
2.1.3 La troisième période : la jeunesse de l'open source....12
2.1.4 Le futur de l'open source....13
2.2 Les dates importantes....13
3 Licence....16
3.1 Licence : Définition....16
3.2 Les principales catégories de logiciels....17
3.2.1 Logiciels propriétaires....17
3.2.2 Logiciels open source / libres....17
3.2.3 Shareware (Partagiciels)....17
3.2.4 Freeware....17
3.2.5 Logiciels privés....17
3.3 Les licences open source / logiciel libre....18
3.3.1 Qu'est ce qu'une licence open source / logiciel libre....18
3.3.2 Quelques licences open source et logiciel libre....19
4 Les formats de fichiers (Formats ouverts)....24
4.1 Les formats ouverts....24
4.2 Les formats fermés....24
4.3 Les formats propriétaires....25
4.4 Les formats standards....25
Chapitre 2 : Les personnes qui font vivre l'Open Source....27
1 Les acteurs du monde libre....27
1.1 Les principaux acteurs de la mouvance Logiciel Libre....27
1.1.1 Richard M. Stallman....27
1.1.2 Free Software Foundation (FSF)....28
1.1.3 GNU Project....28
1.2 Les principaux acteurs de la mouvance Open Source....28
1.2.1 Bruce Perens....28
1.2.2 Eric S. Raymond....29
1.2.3 Tim O'Reilly....29

1.2.4 Open Source Initiative (OSI)..... 29
1.3 Les acteurs non classables dans l'une ou l'autre mouvance..... 29
1.3.1 Linus Torvalds..... 29
1.3.2 Berkeley, Université de Californie..... 29
2 Les communautés Open Source..... 30
2.1 Les communautés..... 30
2.2 Différents types de communautés..... 32
2.2.1 Les communautés de « base »..... 32
2.2.2 Les communautés « sponsorisées »..... 32
2.2.3 Les communautés « issues du privé »..... 32
2.3 Pourquoi rejoindre une communauté Open Source ?..... 33
2.4 Mode de développement des logiciels open source..... 35
2.4.1 La méthode « Cathédrale »..... 35
2.4.2 La méthode « Bazar »..... 35
2.4.3 Avantages et inconvénients de chaque méthode..... 37
Chapitre 3 : Enjeux et risques de l'utilisation de l'Open Source 43
1 Analyse des enjeux stratégiques pour les administrations publiques de passer à l'open source..... 43
1.1 Les enjeux politiques liés aux formats standards ouverts..... 43
1.2 Les enjeux économiques liés aux coûts des licences..... 45
1.3 Les enjeux technologiques liés à l'évolution des logiciels et l'innovation..... 48
1.4 Les enjeux humains liés à l'augmentation des compétences du personnel IT..... 49
1.5 Les enjeux stratégiques liés à la non dépendance vis-à-vis d'un éditeur unique..... 51
1.6 Les enjeux économiques liés aux utilisateurs..... 51
2 Analyse des risques du passage à l'open source par les administrations publiques..... 52
2.1 Les risques de pérennité du logiciel liés au mode de développement de l'open source 52
2.2 Les risques de sécurité liés au mode de développement de l'open source..... 53
2.3 Les risques d'interopérabilité « organisationnelle » liés aux faibles externalités de réseau actuel de l'open source dans le monde des administrations..... 54
2.4 Les risques liés à la maturité des logiciels open source et à la compétence de l'équipe IT..... 54
2.5 Les risques sociaux liés au changement d'environnement de travail..... 55
2.6 Les risques matériels liés au passage à l'open source..... 55
2.7 Les risques logiciels liés à l'interopérabilité technique..... 56
Chapitre 4 : évaluation de l'existant en logiciel open source..... 59
1 Les différentes catégories de logiciels analysées..... 59
1.1 Système d'exploitation ou « Operating system » 59
1.2 Suite bureautique..... 60
1.3 Navigateur Internet..... 60
1.4 Client email ou client de courrier électronique 60
1.5 Groupware ou logiciel collaboratif..... 60
2 Recension des logiciels propriétaires utilisés à la RW et leur équivalent open source..... 60
3 Comparaison des solutions au point de vue technique et financier..... 61
3.1 Système d'exploitation..... 61
3.2 Suite Bureautique..... 62
3.3 Navigateur Internet..... 63
3.4 Client Mail..... 64

3.5 Groupware..... 65
Chapitre 5 : Travail préparatoire à la réalisation d'un guide d'étapes pour une migration vers l'Open Source..... 67
1 Qu'est ce qu'une migration ?..... 67
1.1 Différents types de migrations..... 68
1.1.1 Migration imposée vs Migration volontaire..... 68
1.1.2 Migration en douceur vs Migration rapide..... 69
1.1.3 Migration partielle vs Migration totale..... 70
2 Études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source..... 72
2.1 Expériences Internationales et Nationales..... 72
2.1.1 Internationales..... 72
2.1.2 Nationales..... 76
3 Identification et analyse de guides d'étapes existants..... 81
3.1 Guides d'étapes..... 81
3.1.1 Le guide IDA de migration vers l'Open Source, [Choppy, 2003]..... 82
3.1.2 Managing Linux Migration, [Deb & Sirvastava, 2004]..... 83
3.1.3 Migration vers OpenOffice.org sous Environnement Windows : analyse d'expériences internationales, [Aboubekr & Rivard, 2005]..... 84
3.2 Propositions d'étapes..... 85
Chapitre 6 : Rédaction du guide d'étapes sur base du travail préparatoire..... 87
1 Étapes préliminaires..... 88
1.1 Lancement de la migration..... 88
1.2 Définition des équipes chargées de la migration..... 88
2 Analyse de l'existant matériel..... 89
2.1 Définition et objectifs..... 89
2.2 Contexte d'utilisation de la méthode..... 89
2.3 Les acteurs..... 89
2.3.1 Qui participe à l'analyse de l'existant matériel ?..... 89
2.3.2 Comment recruter et convier les participants ?..... 89
2.4 Mise en oeuvre de la méthode..... 90
2.4.1 Quel est la méthode à utiliser pour réaliser une analyse de l'existant matériel ?..... 90
2.4.2 Quelle est la durée de l'analyse de l'existant Matériel ?..... 90
2.4.3 Quel est le résultat de l'analyse de l'existant matériel ?..... 91
2.5 Atouts et limites..... 91
2.5.1 Atouts..... 91
2.5.2 Limites..... 91
2.5.3 Quand opter pour cette méthode ?..... 91
2.6I nformations complémentaires..... 92
3 Analyse de l'existant logiciel..... 92
3.1 Définition et objectifs..... 92
3.2 Contexte d'utilisation de la méthode..... 92
3.3 Les acteurs..... 93
3.3.1 Qui participe à l'analyse de l'existant Logiciel ?..... 93
3.3.2 Comment recruter et convier les participants ?..... 93
3.4 Mise en oeuvre de la méthode..... 93
3.4.1 Quelle est la durée de l'analyse de l'existant logiciel ?..... 94

3.4.2 Quel est le résultat de l'analyse de l'existant logiciel ?..... 94
3.5 Atouts et limites..... 94
3.5.1 Atouts..... 94
3.5.2 Limites..... 95
3.5.3 Quand opter pour cette méthode ?..... 95
3.6 Informations complémentaires..... 95
4 Proposition de solutions candidates..... 95
4.1 Définition et objectifs..... 95
4.2 Contexte d'utilisation de la méthode..... 96
4.3 Les acteurs..... 96
4.3.1 Qui participe à la proposition de solutions candidates ?..... 96
4.3.2 Comment recruter et convier les participants ?..... 96
4.4 Mise en oeuvre de la méthode..... 96
4.4.1 Faut-il faire appel à un sous-traitant pour réaliser la proposition de solutions candidates ?..... 98
4.4.2 Quelles sont les ressources nécessaires ?..... 98
4.4.3 Combien de participants faut-il prévoir ?..... 98
4.4.4 Quelle est la durée de la proposition de solutions candidates ?..... 98
4.4.5 Quel est le résultat de la proposition de solutions candidates ?..... 98
4.5 Atouts et limites..... 98
4.5.1 Atouts..... 98
4.5.2 Limites..... 99
4.5.3 Quand opter pour cette méthode ?..... 99
5 Analyse technique et fonctionnelle..... 99
5.1 Définition et objectifs..... 99
5.2 Contexte d'utilisation de la méthode..... 100
5.3 Les acteurs..... 100
5.3.1 Qui participe à l'analyse technique ?..... 100
5.4 Mise en oeuvre de la méthode..... 100
5.4.1 Faut-il faire appel à un sous-traitant pour réaliser une analyse technique et fonctionnelle ?..... 100
5.4.2 Quelles sont les ressources nécessaires ?..... 101
5.4.3 Combien de participants faut-il prévoir ?..... 101
5.4.4 Quelle est la durée de l'analyse technique et fonctionnelle ?..... 101
5.4.5 Quel est le résultat de l'analyse technique et fonctionnelle ?..... 101
5.5 Atouts et limites..... 101
5.5.1 Atouts..... 101
5.5.2 Limites..... 101
5.5.3 Quand opter pour cette méthode ?..... 101
6 Analyse économique..... 102
6.1 Définition et objectifs..... 102
6.2 Contexte d'utilisation de la méthode..... 102
6.3 Les acteurs..... 102
6.3.1 Qui participe à l'analyse économique ?..... 102
6.3.2 Comment recruter et convier les participants ?..... 104
6.4 Mise en oeuvre de la méthode..... 104
6.4.1 Faut-il faire appel à un sous-traitant pour réaliser une l'analyse économique ? 104

6.4.2 Quelles sont les ressources nécessaires ?....104
6.4.3 Comment préparer une analyse économique ?....104
6.4.4 Combien de participants faut-il prévoir ?....104
6.4.5 Quelle est la durée de l'analyse économique ?....104
6.4.6 Quel est le résultat de l'analyse économique ?....104
6.5 Atouts et limites....105
6.5.1 Atouts....105
6.5.2 Limites....105
6.5.3 Quand opter pour cette méthode ?....105
6.6 Informations complémentaires....105
7 Choix de la solution finale....105
7.1 Définition et objectifs....105
7.2 Contexte d'utilisation de la méthode....106
7.3 Les acteurs....106
7.3.1 Qui participe au choix de la solution finale ?....106
7.3.2 Comment recruter et convier les participants ?....106
7.4 Mise en oeuvre de la méthode....106
7.4.1 Quelle est la durée du choix de la solution finale ?....106
7.4.2 Quel est le résultat du choix de la solution finale ?....106
7.5 Atouts et limites....107
8 Présentation de la nouvelle solution....107
8.1 Définition et objectifs....107
8.2 Contexte d'utilisation de la méthode....107
8.3 Les acteurs....107
8.3.1 Qui participe à la présentation sur la nouvelle solution ?....107
8.3.2 Comment recruter et convier les participants ?....108
8.4 Mise en oeuvre de la méthode....108
8.4.1 Faut-il faire appel à un sous-traitant pour réaliser la présentation de la nouvelle solution ?....108
8.4.2 Quelles sont les ressources nécessaires ?....108
8.4.3 Combien de participants faut-il prévoir ?....108
8.4.4 Quelle est la durée de la présentation de la nouvelle solution ?....108
8.4.5 Quel est le résultat de la présentation de la nouvelle solution ?....108
8.5 Atouts et limites....109
8.5.1 Atouts....109
8.5.2 Limites....109
8.5.3 Quand opter pour cette méthode ?....109
9 Formation de l'équipe IT....109
9.1 Définition et objectifs....109
9.2 Les acteurs....109
9.3 Mise en oeuvre de la méthode....110
9.3.1 Faut-il faire appel à un sous-traitant pour réaliser un la formation de l'équipe IT ?....110
9.3.2 Combien de participants faut-il prévoir ?....110
9.3.3 Quelle est la durée de la formation de l'équipe IT ?....110
9.3.4 Quel est le résultat de la formation de l'équipe IT ?....110
9.4 Atouts et limites....110
9.4.1 Atouts....110

9.4.2 Limites..... 110
9.4.3 Quand opter pour cette méthode ?..... 111
10 Tests d'intégration de la nouvelle solution..... 111
10.1 Définition et objectifs..... 111
10.2 Contexte d'utilisation de la méthode..... 111
10.3 Les acteurs..... 111
10.3.1 Qui participe aux tests d'intégration de la nouvelle solution ?.....111
10.3.2 Comment recruter et convier les participants ?.....112
10.4 Mise en oeuvre de la méthode..... 112
10.4.1 Faut-il faire appel à un sous-traitant pour réaliser les tests d'intégration de la nouvelle solution ?..... 112
10.4.2 Quelles sont les ressources nécessaires ?.....112
10.4.3 Comment préparer les tests d'intégration de la solution finale ?..... 112
10.4.4 Quelle est la durée des tests d'intégration de la nouvelle solution ?..... 112
10.4.5 Quel est le résultat des tests d'intégration de la nouvelle solution ?..... 112
10.5 Atouts et limites.....113
10.5.1 Atouts.....113
10.5.2 Limites..... 113
10.5.3 Quand opter pour cette méthode ?..... 113
11 Formation des utilisateurs..... 113
11.1 Définition et objectifs..... 113
11.2 Contexte d'utilisation de la méthode..... 113
11.3 Les acteurs..... 114
11.3.1 Qui participe à la formation des utilisateurs ?..... 114
11.3.2 Comment recruter et convier les participants ?.....114
11.4 Mise en oeuvre de la méthode..... 114
11.4.1 Faut-il faire appel à un sous-traitant pour réaliser la formation des utilisateurs ?..... 114
11.4.2 Quelles sont les ressources nécessaires ?.....114
11.4.3 Comment préparer la formation des utilisateurs ?..... 114
11.4.4 Combien de participants faut-il prévoir ?..... 115
11.4.5 Quelle est la durée de la formation des utilisateur ?..... 115
11.4.6 Quel est le résultat de la formation des utilisateurs ?.....115
11.5 Atouts et limites.....115
11.5.1 Atouts.....115
11.5.2 Limites..... 115
11.5.3 Quand opter pour cette méthode ?..... 115
12 Mise en place de la nouvelle solution..... 116
12.1 Définition et objectifs..... 116
12.2 Contexte d'utilisation de la méthode..... 116
12.3 Les acteurs..... 116
12.3.1 Qui participe à la mise en place de la nouvelle solution ?..... 116
12.3.2 Comment recruter et convier les participants ?.....116
12.4 Mise en oeuvre de la méthode..... 117
12.4.1 Faut-il faire appel à un sous-traitant pour réaliser la mise en place de la nouvelle solution ?..... 117
12.4.2 Quelles sont les ressources nécessaires ?.....117
12.4.3 Comment préparer la mise en place de la nouvelle solution ?.....117

12.4.4 Combien de participants faut-il prévoir ?..... 117
12.4.5 Quelle est la durée de la mise en place de la migration ?..... 118
12.4.6 Quel est le résultat de la mise en place de la nouvelle solution ?..... 118
12.5 Quand opter pour cette méthode ?..... 118
12.6 Informations complémentaires..... 118
Conclusion..... 119
Bibliographie..... 121
1 Livres..... 121
2 Articles..... 121
3 Articles sur Internet..... 122
4 Sites Internet Génériques..... 123
Annexes..... 125
1 The Open Source Definition..... 125
2 Licences..... 128
2.1 The GNU General Public License (GPL)..... 128
2.2 Open Software License ("OSL") v. 3.0..... 134
2.3 Mozilla Public License 1.1 (MPL 1.1)..... 138
2.4 The BSD License..... 146
2.5 The MIT License..... 147
2.6 Apache License, Version 2.0..... 148
2.7 RECIPROCAL PUBLIC LICENSE..... 152
3 Questionnaire distribué au personnel d'EASI-WAL..... 162
4 Processus de Migration du guide IDA, [Choppy, 2003]..... 166

Liste des acronymes

ASBL : Association Sans But Lucratif

AT&T : American Telephone and Telegraph Company

AWT : Agence Wallonne des Télécommunications

BSD : Berkley Software Distribution

CEO : Chief Executive Officer

CMBD : Configuration Management DataBase

CVS : Concurrent Versions System

EASI-WAL : Commissariat de la Région Wallonne à l'E-Administration et à la Simplification

FedICT : SPF Technologie de l'Information et de la Communication

FSF : Free Software Fundation

GNOME : GNU Network Object Model Environment

GNU : GNU's Not Unix

GPL : General Public License

IADBC : Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens

IDA : Interchange of Data between Administrations

IE : Internet Explorer

IT : Information Technology

ITIL : Information Technology Infrastructure Library

KDE : K Desktop Environment

LGPL : Library or Lesser General Public License

MIT : Massachusetts Institute of Technology

MPL : The Mozilla Public License

NTT : Nippon Télégraphe et Téléphone

OSDL : Open Source Development Lab

OSI : Open Source Initiative

OSL : Open Software License

PME : Petites et Moyennes Entreprises

RMS : Richard M. Stallman

ROI : Return On Investment

RPL : Reciprocal Public License

TCO : Total Cost of Ownership

Introduction

Les administrations publiques se posent à l'heure actuelle la question de l'opportunité de migrer, tout ou une partie de leur système informatique, à l'Open Source. Cette question est légitime vu l'essor que prennent les logiciels qui se cachent derrière ce terme. Mais elle recèle encore beaucoup d'incertitudes. Ces dernières concernent principalement :

- le manque de connaissances des différentes solutions existantes,
- le développement à long terme des logiciels Open Source et leur maturité,
- la pérennité des acteurs présents et l'étendue du marché couvert par les logiciels Open Source,
- les méthodes possibles pour migrer leur système informatique.

C'est pourquoi la Région Wallonne nous a demandé d'analyser ces questions. Ce mémoire a donc pour objectif d'y répondre au mieux.

La première partie sera consacrée à répondre à la question : « Qu'est-ce que l'Open Source? ». La réponse reprendra donc les différents aspects de ce qu'est l'Open Source à savoir : sa définition, sa philosophie, son histoire, les licences Open Source, les formats de fichiers ouverts, ses principaux acteurs et les communautés Open Source.

La seconde partie nous amènera à développer différents aspects liés à la migration vers l'Open Source. Nous aborderons donc les enjeux et les risques d'un passage à l'Open Source, nous analyserons quelques cas de migrations réussies et certaines solutions Open Source intéressantes, nous proposerons une définition de ce qu'est une migration et, finalement, nous élaborerons un guide d'étapes pour aider à réaliser une migration.

Ces différents points montrent que ce mémoire ne traite donc pas de développements informatiques et encore moins de technologie. Mais, en tant qu'informaticien, il nous appartient aussi de guider les organisations qui nous emploient dans ce genre de démarches.

Partie 1

Open Source

Chapitre 1 : L'Open Source se découvre

Ce chapitre a pour but de clarifier ce qu'est la philosophie Open Source, d'où elle vient et ce qu'elle apporte comme nouveautés au niveau juridique et au niveau des formats de fichiers.

Il est divisés en quatre points qui vont essayer d'atteindre cet objectif :

- premièrement, planter le décor via quelques définitions,
- deuxièmement, retracer l'histoire de l'Open Source depuis les débuts de l'informatique moderne à nos jours,
- troisièmement, présenter les licences logiciels et plus principalement les licences libres
- et quatrièmement, définir les différents types de formats de fichiers disponibles sur le marché.

1 Définitions

La première définition fut celle de « Logiciel Libre ». Elle fut proposée par la FSF[1] (*Free Software Fundation*) durant le milieu des années 80. Cette définition détermine autant les caractéristiques des licences logiciels que la philosophie sous-jacente aux logiciels libres. Elle est actuellement prônée par la FSF et le projet GNU. Elle fut à la base de la licence GPL. Par la suite, certains estimèrent que le terme « Logiciel Libre » (« free software ») était trop ambigu. En effet, le mot « free » en anglais signifie tant libre que gratuit. Ce qui porte à confusion auprès du grand public. Le terme « Open Source » fut alors proposé fin des années 90. Bruce Perens en proposa une définition basée sur *The Debian Free Software Guidelines* et elle est actuellement à la base d'un label. Le label OS[2]I est décerné par l'*Open Source Initiative* à tout logiciel qui respecte la définition de l'open source.

1 Le site Internet de la FSF : http://www.fsf.org
2 Le site Internet de l'OSI : http://www.opensource.org

Figure 1: Logo de l'Open Source Initiative

Figure 2: Logo de la Free Software Fundation Europe

1.1 Logiciel Libre (« Free Software »)

D'après le site officiel du projet GNU[3], l'expression « Logiciel Libre » se définit par quatre libertés :

- La liberté d'exécuter le programme pour tous les usages (liberté 0).
- La liberté d'étudier le fonctionnement du programme, et de l'adapter à vos besoins (liberté 1).
- La liberté de redistribuer des copies, donc d'aider votre voisin (liberté 2).
- La liberté d'améliorer le programme et de publier vos améliorations, pour en faire profiter toute la communauté (liberté 3).

Pour respecter ces différentes libertés, l'accès au code source est une condition nécessaire. Nous sommes donc libre de redistribuer gratuitement, ou pas, tout logiciel libre en notre possession. Nous pouvons aussi le modifier si nous le désirons.

Le terme « Logiciel Libre » (« free ») signifie liberté non pas gratuité. Richard M. Stallman a une expression bien à lui pour expliquer simplement ce terme « To understand the concept, you should think of *free* as in *free speech* not as in *free beer* ». Donc si nous désirons vendre un logiciel libre, nous en avons tout à fait le droit. Et les initiateurs de la tendance « libre » vont même jusqu'à le conseiller. En effet, ils estiment que la vente permet de fournir des fonds aux développeurs de logiciels libres. Il faut toutefois respecter les règles qui régissent le commerce national et international.

1.2 Open Source

La définition officielle du terme « Open Source » en est à la version 1.9. Elle se compose de 10 aspects fondamentaux[4] :

1. Libre redistribution : la licence ne doit pas empêcher la vente ou le don du logiciel. Et, elle ne doit pas exiger que la vente soit soumise à des droits d'auteurs ou de royalties.
2. Code source : le code source d'un logiciel doit être facilement trouvable, modifiable et lisible.
3. Travaux dérivés : la licence doit autoriser les modifications et les travaux dérivés, et leur distribution avec la même licence que celle d'origine.

3 http://www.gnu.org

4 Veuillez vous reporter dans l'annexe pour la définition complète et officielle de ces différents aspects.

4. Intégrité du code source de l'auteur : certaines licences demandent que les modifications apportées au programme original se fassent uniquement via des patchs ou que les programmes modifiés portent un autre nom que celui d'origine. Cela pour protéger la réputation du programmeur initial.
5. Pas de discrimination entre les personnes et les groupes : la licence ne peut exclure certains utilisateurs potentiels à cause de leur origine, de leur appartenance à un groupe, etc.
6. Pas de discrimination entre les domaines d'application : la licence ne doit pas limiter le champ d'applications d'un logiciel (commercial, bénévole …).
7. Distribution de la licence : il n'est pas nécessaire de se conformer à des licences complémentaires.
8. La licence ne doit pas être spécifique à un produit : les droits liés à un programme sont liés à ce programme et ne dépendent pas de la méthode utilisée pour le distribuer ou du contexte dans lequel il est utilisé (au sein d'une distribution ou seul).
9. La licence ne doit pas contaminer d'autres logiciels : un logiciel inclus dans un groupe de logiciels ne doit pas influencer leur licence.
10. La licence doit être indépendante de la technologie : la licence ne doit pas contraindre le programmeur à utiliser une technologie plutôt qu'une autre. Cela limiterait la réutilisation du code dans d'autres domaines.

Nous pouvons constater que cette définition respecte les quatre libertés de la définition de « Logiciel Libre ». La liberté 0 correspond aux points 5 et 6, la liberté 1 au point 2, la liberté 2 au point 1 et la liberté 3 au point 3. Cela prouve bien que les fondements de ces deux définitions sont identiques.

Nous pouvons aussi constater que par certains points, comme le point 4 qui impose une protection pour la réputation de l'auteur d'origine, cette définition est plus contraignante que celle des logiciels libres. Ces nouvelles contraintes ne vont toutefois pas à l'encontre de la définition des logiciels libres mais peuvent amener à ce qu'un logiciel qualifié d'open source ne puisse pas être qualifié de logiciel libre. Nous proposerons un exemple de ce genre de situations lorsque nous aborderons le point consacré aux licences des logiciels. En effet, pour dire qu'un logiciel est open source et/ou libre, la licence sous laquelle il est proposé doit respecter les définitions ci-dessus.

Nous pouvons donc dire que les logiciels libres sont inclus dans les logiciels open source, Figure 3.

1.3 Logiciel Libre vs Open Source

La Figure 3 reprend une division des logiciels en deux grandes catégories.

- La première catégorie est composée des logiciels propriétaires. Ce sont des logiciels qui ne respectent pas les définitions citées ci-dessus. L'AWT[5] en propose la définition suivante :

  « U*n logiciel propriétaire est un logiciel vendu sous forme de code exécutable accompagné d'une licence qui régit précisément les conditions légales d'utilisation de*

5 [AWT, 2005]

celui-ci ».

Leurs caractéristiques principales sont la confidentialité du code source, l'achat de la licence conditionnant l'utilisation du logiciel, l'interdiction de modifier et redistribuer le logiciel, etc.

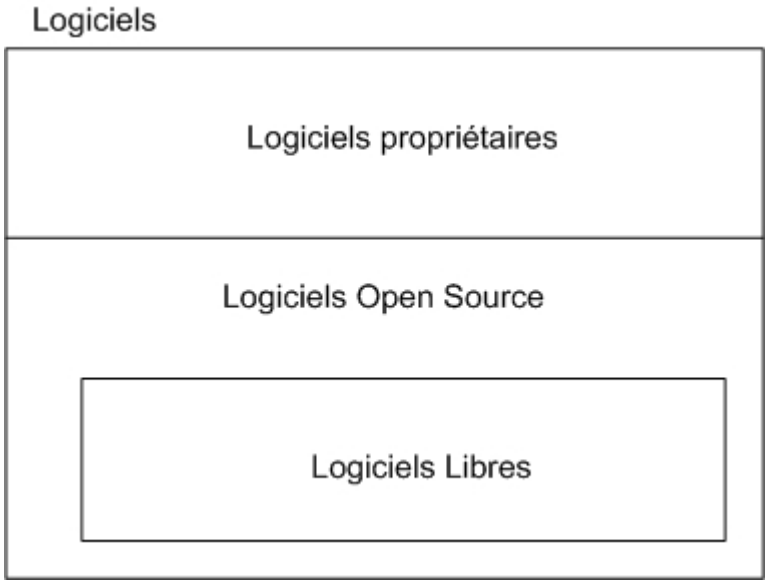


Figure 3: Classification élémentaire des logiciels

- La deuxième catégorie reprend les logiciels open source et les logiciels libres. Comme expliqué ci-dessus, les logiciels libres peuvent être considérés comme étant un sous-ensemble des logiciels open source. D'après les sites Internet de référence[6], la grande majorité des logiciels open source sont des logiciels libres et la différence est si ténue que nous pouvons aisément utiliser indifféremment les deux termes sans risque d'erreurs majeures.

  Néanmoins, dans la suite de ce mémoire, je vais principalement utiliser le terme « Open Source » comme terme générique et j'utiliserai les termes précis quand la distinctions entre open source et logiciel libre est nécessaire.

1.4 Philosophie

Pourquoi ces deux définitions qui sont finalement assez proches ne pourraient-elles pas fusionner pour donner naissance à une seule définition qui conviendrait à tout le monde ? Pour répondre à cette question, il faut se plonger dans les philosophies qui dominent chaque définition.

- Le clan des logiciels libres défend l'idée de la liberté. Et estime, par ailleurs, que le terme « Open Source » ne prône que l'idée du code source ouvert et pas les autres points de sa définition.
- Le clan des logiciels open source, prône l'action et l'utilisation des logiciels open source en les comparant aux logiciels propriétaires. En outre ils estiment que le terme « Free

6 Les sites Internet de références sont ici ceux de l'OSI et de la FSF.

Software » ne désigne auprès du grand publique que la gratuité et non la liberté prônée par ce terme.

Nous voyons donc qu'il est actuellement impossible de réunir ces deux clans dans un même groupe car il n'y a pas moyen de trouver un terme unique qui désigne correctement les logiciels qu'ils défendent et que leur façon d'agir est totalement différente. D'un coté, ils essayent de jouer sur le tableau de la philosophie, en prônant la liberté, tandis que de l'autre, ils jouent sur le tableau de la réalité économique en proposant le label OSI, en faisant de la publicité pour leur label et en prouvant la capacité technique de leurs logiciels.

1.5 Utilisation des logiciels open source

De par sa définition, nous voyons clairement que tout le monde peut utiliser les logiciels open source. Quel que soit son domaine : utilisation privée, publique ou commerciale.

Les logiciels open source ne sont pas limités au système d'exploitation GNU/Linux. En effet, un grand nombre de logiciels open source sont développés pour fonctionner sous différents systèmes d'exploitation : GNU/Linux, Windows, Mac, etc.

Cette optique de développement leur permet d'ailleurs d'avoir une plus grande visibilité auprès du grand public qui utilise en majorité Windows. Par exemple, au commissariat EASI-WAL[7], un ensemble de logiciels open source (Firefox, Thunderbird, Phprojekt et OpenOffice) est utilisé fonctionnant sous Windows XP.

2 Histoire de l'open source

Nous allons scinder l'histoire de l'informatique et principalement celle des logiciels libres et open source en deux parties.

La première sera consacrée à un résumé succinct de l'histoire de l'open source divisée en trois périodes. Cette division en périodes, bien que différente sous certains aspects, est inspirée principalement de [Raymond, 2001] et de [Josh Lerner and Jean Triole].

Dans la seconde partie, j'énumérerai des dates qui méritent d'être retenues. Ces dates ont été reprises d'articles, de sites Internet et de livres référencés dans la bibliographie.

2.1 L'Histoire

L'histoire de l'informatique libre est généralement divisée en trois périodes. La première période, va du début de l'informatique au début des années 80, où le partage du code source était chose courante. La deuxième période va du début des années 80 jusqu'au milieu des années 90, c'est-à-dire du début des logiciels propriétaires à la sortie de la version1.0 de Linux. La troisième et dernière période est celle qui va du milieu des années 90 jusqu'à nos jours. Cette période a vu la création et la structuration du monde libre comme on le connaît actuellement.

On pourrait dire que nous sommes actuellement dans le début d'une quatrième période qui voit l'arrivée d'une certaine maturité des logiciels open source. Mais seul l'avenir nous dira si cette nouvelle période a lieu d'être.

7 EASI-WAL : Commissariat à l'e-gouvernement et à la simplification administrative de la Région Wallonne,

2.1.1 La première période : la préhistoire de l'informatique

Au début l'informatique était surtout le fait du milieu scientifique et des chercheurs qui essayaient de mettre en place de nouvelles technologies. Le partage des informations était donc nécessaire pour progresser. Avec l'apparition d'ordinateurs de plus en plus puissants, le développement de logiciels va progresser de manière considérable. On voit apparaître les premiers systèmes d'exploitation, les premiers compilateurs, les premiers langages de programmation et donc les premières applications.

En 1969, la mise au point de l'ARPAnet, premier réseau qui reliait la défense américaine, des équipes de chercheurs universitaires et des laboratoires militaires, permit la mise en place d'une communauté générale des hackers[8].

Cette année coïncide aussi avec la création de Unix par Ken Thompson, un hacker des laboratoires Bell. Unix est développé en C.

En 1978, tout l'environnement développé autour d'Unix est porté sur des ordinateurs de différents types. C'est l'avènement des systèmes d'exploitations multi-plateformes. Ce fut une révolution. En effet, les chercheurs ne devront plus redévelopper tous leurs outils chaque fois qu'ils changent de plateforme. Unix fut adopté par la plupart des chercheurs.

1980 marque l'apparition de Usenet, un réseau non militaire. Il va permettre aux hackers de développer des sites Usenet qui permettront l'échange de données.

En 1975, le premier ordinateur personnel fut commercialisé. En 1976, Apple fut fondé et connu un succès incroyable. Ces deux événements ont permis à une nouvelle génération de se développer. Leur langage n'est pas le C mais le BASIC.

2.1.2 La deuxième période : la naissance de l'open source

Vers 1980, AT&T commence à faire valoir ses droits de propriété intellectuelle sur Unix. En 1985, Richard Stallman, du MIT, crée la Free Software Fondation qui va développer le projet GNU (GNU's not Unix), lancé en 1983.

Le projet GNU a pour objectif de créer un Unix like totalement libre. La plupart des logiciels développés par ce projet seront mis sous la licence GPL (General Public License) qui fut lancée en 1989. C'est durant cette période que va apparaître Microsoft et que les logiciels propriétaires vont s'imposer sur les premiers ordinateurs personnels.

C'est également à ce moment que la véritable croissance des logiciels libres va débuter avec le lancement en 1991 de Linux et de la première distribution, Debian, en 1993.

2.1.3 La troisième période : la jeunesse de l'open source

Le 12 mars 1994, la sortie de la version 1.0 de Linux annonce l'explosion du développement des logiciels open source. Malheureusement, Microsoft prend, au même moment, la place de leader incontesté qu'on lui connaît encore actuellement.

8 Différence entre hacker et cracker : un hacker est une personne qui programme et développe pour le plaisir. Il essaye de se dépasser en trouvant des solutions innovantes à des problèmes qu'il rencontre. Être un hacker est une façon de vivre, une philosophie. Un cracker est une personne qui « pirate » des données accessibles sur Internet. Dans les médias, on utilise souvent le mot « hacker » en lieu et place du mot « cracker ».

Durant cette période, les succès s'enchaînent aussi pour les logiciels open source : une pléthore de distributions apparaît, Netscape livre son navigateur à la communauté, et Sun Mycrosystem le suit en donnant le code de sa suite bureautique StarOffice.

Les logiciels open source prennent petit à petit une place importante lors du choix de solutions informatiques. La concurrence avec Microsoft est réelle. L'open source commence sa croisade de séduction avec des logiciels matures comme la version 2.0 d'OpenOffice.org, la version 1.5 de Mozilla Firefox, Apache, etc.

Les distributions basées sur Linux à destination du grand public débarquent en force comme Red Hat, SuSe et plus récemment Ubuntu et Kubuntu[9].

Et finalement, les majors du secteur investissent et proposent des solutions professionnelles basées sur Linux (IBM[10], HP, etc.).

2.1.4 Le futur de l'open source

Le futur de l'open source commence avec l'arrivée en force de logiciels matures et de sociétés qui proposent des services et solutions basées sur les logiciels open source. Ils vont maintenant devoir montrer leur viabilité à long terme.

L'utilisation des standards ouverts étant aussi une priorité des différentes administrations de par le monde, elle va permettre à l'open source de se distinguer par rapport à Microsoft qui ne les propose pas dans ses logiciels.

Pour conclure, on peut dire que l'open source entre dans une aire de maturité, où les défis à venir vont lui permettre de voir si sa philosophie et sa méthode de développement tiennent la route sur le long terme face à des solutions qui ont déjà fait leurs preuves malgré leurs forces et leurs faiblesses.

2.2 Les dates importantes

- 27 septembre 1983 : Richard M. Stallman annonce le lancement le projet GNU.
- 1985 : création de la Free Software Foundation.
- 1986 : création du langage Perl[11] par Larry Wall.
- 1989 : la FSF écrit la première version de la GNU General Public licence.
- 25 août 1991 : la création de Linux est annoncée par Linus Torvald sur internet.
- 1993 : création de la distribution Debian par Ian Murdock.
- 12 mars 1994 : sortie de Linux 1.0.
- 1994 : création des distributions SuSe et Red Hat.
- Avril 1995 : sortie de la première version officielle du serveur Apache.

9 Le site Internet de ces différentes distributions se trouve dans la bibliographie.
10 Le site de référence Linux de IBM est : http://www-1.ibm.com/linux.
11 Perl (*Practical Extraction and Report Language)* est un langage interprété. Pour plus d'informations, se référer au site Internet officiel référencé dans la bibliographie.

Histoire de l'open source

- 8 juin 1996 : sortie de la version 2.0 de Linux, le nombre d'utilisateurs est estimé alors à 1,5 millions.
- 4 octobre 1995 : création de l'environnement KDE.
- 1995 : le Php[12] est créé par Rasmus Lerdorf.
- Aout 1997 : création du projet GNOME en concurrence au projet KDE qui n'est pas libre, car utilise la librairie Qt de Trolltech qui n'était pas encore libre.
- 1998 : création de l'Open Source Initiative.
- 1998 : création du projet Mozilla, grâce à Netscape qui rend une partie de son code source libre.
- 1998 : création de la distribution Mandrake.
- 1999 : création de la première distribution sous format de live CD.
- Septembre 2000 : la version 2.2 de Qt est mise sous licence GPL.
- Octobre 2000 : sortie de la version 2.0 de KDE qui est maintenant totalement libre, grâce à Qt 2.2.
- 13 octobre 2000 : Sun Microsystems donne à la communauté libre le code source de la suite bureautique StarOffice 5.2. Son nom de projet sera OpenOffice.org.
- Décembre 2003 : Apache, serveur Internet, dépasse la barre des 60% d'utilisation tous systèmes d'exploitation confondues. La Figure 4[13] montre bien l'évolution d'Apache au fil des ans.

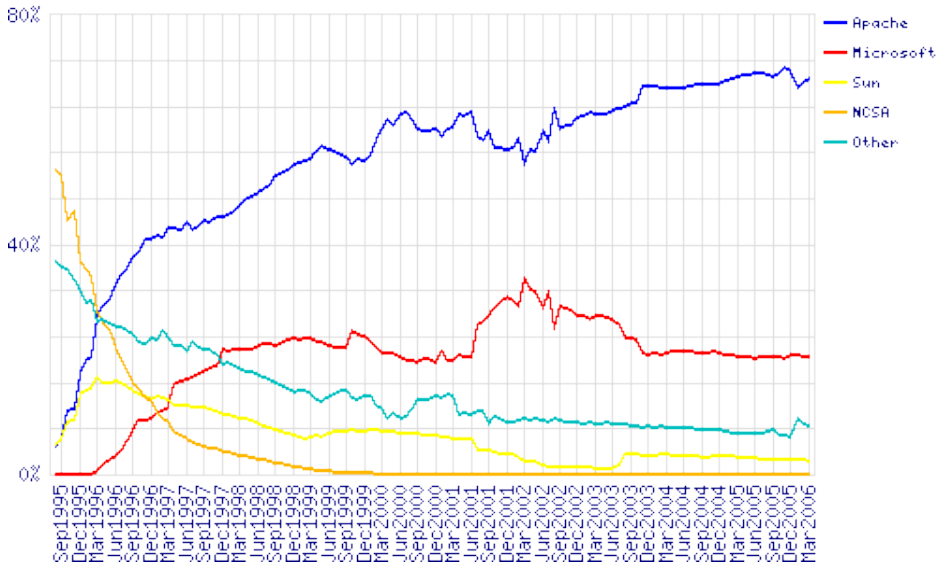


Figure 4: Répartition du marché des serveurs principaux pour tous les domaines Août 1995 - Mars 2006, selon le site Internet Netcraft.

- Octobre 2004 : sortie de la première version de la distribution Ubuntu.
- 20 octobre 2005 : sortie de la version 2.0 de OpenOffice.org qui sera plébiscitée par un

12 Pour en savoir plus sur l'histoire de php, veillez visiter le site Internet suivant : http://fr.php.net/history.
13 Image tirée du site Internet Netcraft.

grand nombre. D'après les membres de l'organisation[14], OpenOffice.org possède environ 10% des parts du marché des suites bureautiques.

- 29 novembre 2005: sortie de Mozilla Firefox 1.5. Selon une étude de Xiti Monitor[15], on peut dire que Firefox possède plus de 20% des parts de marché des navigateurs en Europe.

  La Figure 5 nous montre bien la pénétration en Europe de Firefox. Et sur la Figure 6, nous voyons clairement que les parties du monde ayant un accès facile à Internet sont celles qui ont la plus forte pénétration de Firefox.

- 12 janvier 2006 : sortie de la version 1.5 de Mozilla Thunderbird.

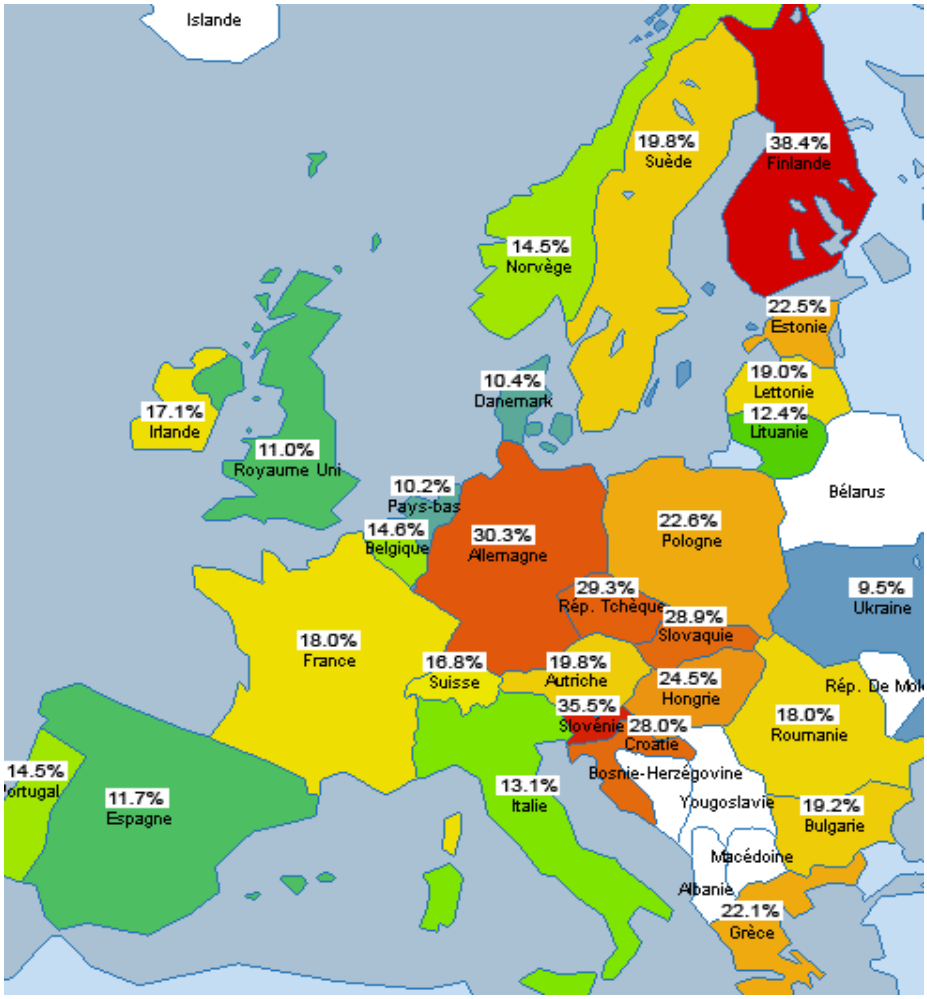


Figure 5: Pénétration de Firefox au niveau Européen selon Xiti Monitor

14 Informations recueillies sur le site Internet suivant : http://www.vnunet.fr/actualite/logiciels/utilitaires/20051021001.

15 Étude réalisée le 08/01/06 sur un échantillon de sites Internet professionnels audités par XiTi, sur un total de 32 544 568 visites.

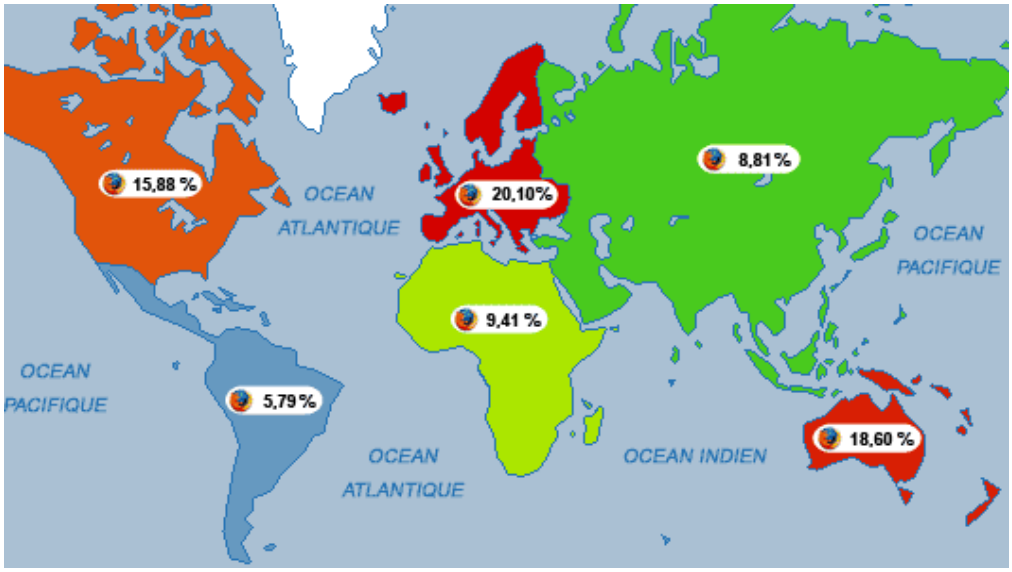


Figure 6: Utilisation de Firefox au niveau mondiale selon Xiti Monitor

3 Licence

Avant de décrire avec précision les plus importantes licences open source, il est important de comprendre ce qu'est une licence en général et que c'est principalement la licence qui détermine la catégorie à laquelle appartient un logiciel.

3.1 Licence : Définition

Nous allons commencer par citer quelques définitions :

- « *Une **licence** est un contrat par lequel le titulaire des droits d'un logiciel autorise un tiers à poser des gestes qui autrement les enfreindraient.* » Pierre-Paul Lemeyre, *Les logiciels libres sous l'angle de la Responsabilité Civile*, 2002
- « *Une **licence** est un document ou un contrat donnant le droit de faire quelque chose* »[16]

Dans ces deux définitions et dans les autres que nous pouvons trouver, nous remarquons donc qu'une licence est un contrat dans le sens juridique du terme. Et que les licences logiciels sont étroitement liées au droit d'auteur.

Le droit d'auteur concerne :

- la paternité de l'oeuvre,
- les droits liés à la distribution de l'oeuvre,
- les droits liés à la commercialisation de l'oeuvre,
- les droits liés à la modification de l'oeuvre,
- etc.

16 Cette définition a été reprise du site Internet Wikipedia.

Nous verrons par la suite que c'est par l'utilisation des droits d'auteur que les licences open source et les logiciels libres se distinguent des autres licences et plus particulièrement des licences propriétaires.

3.2 Les principales catégories de logiciels

Les différentes catégories reprises ci-dessous sont tirées de l'article *Catégories de logiciels libres et non libres* disponible sur le site Internet de la FSF. Nous allons donc définir de manière précise et non ambiguë les plus courantes de ces catégories.

3.2.1 Logiciels propriétaires

Pour reprendre ce qui a déjà été dit à leur sujet dans le premier point de ce chapitre, nous dirons donc qu'un logiciel propriétaire est un logiciel livré sous forme de code exécutable et dont l'utilisation est fortement réglementée par la licence qui l'accompagne. Il ne peut généralement pas être copié, modifié ou redistribué sans l'obtention d'une licence complémentaire.

La licence qui les accompagne est une licence où peu de droits sont donnés aux licenciés, ici les utilisateurs.

3.2.2 Logiciels open source / libres

Un logiciel open source ou un logiciel libre est un logiciel dont la licence respecte la définition d'open source ou de logiciel libre donnée dans le premier point de ce chapitre.

Dans la suite, nous verrons que la majorité des licences correspondant à la définition de logiciel open source correspondent aussi à la définition de logiciel libre. Et que peu de licences open source ne sont pas des licences logiciel libre. Ce qui confortera notre idée selon laquelle il n'est pas aisé et souvent peu utile de différencier open source et logiciel libre.

3.2.3 Shareware (Partagiciels)

Les shareware sont des logiciels dont la redistribution est autorisée mais pas la modification.

Toute personne, qui continue à utiliser une copie après un certain temps, souvent de 10 à 30 jours, est obligée de payer des royalties à l'auteur du logiciel.

3.2.4 Freeware

Terme désignant les logiciels dont l'utilisation et la redistribution sont autorisées mais pas la modification. Cette restriction est aisément respectable car leur code source est rarement disponible. Ce qui les distingue des logiciels open source.

3.2.5 Logiciels privés

Comme son nom l'indique, un logiciel privé est destiné à rester inconnu du public. Il est développé par une personne, une société, une organisation pour ses besoins propres.

Les licences des logiciels privés vont des licences propriétaires à des licences proches des licences open source mais où la liberté de redistribution est fortement limitée.

3.3 Les licences open source / logiciel libre

3.3.1 Qu'est ce qu'une licence open source / logiciel libre

Maintenant que nous avons défini ce qu'est une licence, que nous savons ce qu'est un logiciel open source et un logiciel libre, nous allons examiner en détail les caractéristiques propres aux licences open source et aux licences libres.

Comme dit précédemment, une licence est un contrat, ici le contrat lie un groupe de développeurs open source à l'utilisateur du logiciel. Il s'agit donc d'un contrat entre deux parties.

Nous entendons souvent dire que les licences open source vont à l'encontre du droit d'auteur. Cela est totalement faux. Ces licences respectent le droit d'auteur mais d'une façon différente. En effet, le droit d'auteur accorde au créateur le droit d'autoriser et d'interdire la reproduction, la modification, la distribution, etc.

Pour mieux comprendre cette explication, nous allons reprendre quelques phrases de [Etienne Montero] qui expriment clairement la vision des licences open source par rapport au droit d'auteur :

« Dans le modèle libre, l'emphase est mise sur ce second aspect [autorisation] du droit d'auteur. Ce pouvoir d'autoriser et d'interdire est utilisé ici, moins pour jouir des droits de manière privative et procurer une rémunération, que pour promouvoir la liberté de création et de diffusion, en accordant au licencié des droits patrimoniaux extrêmement larges. Cela étant, d'abord, l'auteur prend appui sur ses droits d'auteur pour concéder des droits étendus sur son logiciel; ensuite, il entend invoquer ses droits économiques au cas où un preneur de licence n'en respecterait pas les termes; enfin, il n'hésitera pas à faire valoir ses droits moraux, qu'il a conservés, en cas d'atteinte portée à ceux-ci. »

Nous voyons donc qu'une licence open source respecte, de manière un peu originale, le droit d'auteur.

Une licence open source ou libre est une licence qui reprend en son sein les différentes libertés exprimées par leur définition. Donc une licence open source ou libre doit autoriser :

- l'utilisation,
- la reproduction,
- la diffusion,
- la modification,

Le droit d'auteur permettant d'accorder ces libertés.

Certaines licences open source font aussi appel au « copyleft » qui a une influence sur les travaux dérivés. Nous pouvons aussi dire que le copyleft est utilisé avec différentes puissances : le copyleft fort de la GPL, le copyleft normal de la OSL et le copyleft faible de la MPL.

La majorité des licences open source ont aussi deux types de clauses :

- la « Copyright notice » qui reprend le nom de l'auteur ou des auteurs. Cette clause est liée à l'importance, notamment, du droit de paternité en matière d'open source.

- la clause de non-garantie qui est liée au fait que l'auteur accorde beaucoup de droits sans aucune contrepartie (financière). Cela n'empêche pas un tiers de proposer une garantie indépendante.

3.3.2 Quelques licences open source et logiciel libre

Nous ne pouvions pas clôturer ce point sur les licences sans présenter quelques licences open source et/ou libre. En effet, c'est principalement par les licences acceptées par l'une ou l'autre définition que nous pouvons distinguer les deux termes.

La Figure 3, nous montrait que les logiciels libres étaient inclus dans les logiciels open source. Nous allons donc commencer par présenter les licences de logiciel libre les plus connues et les plus utilisées. Puis nous présenterons une licence open source mais non libre. Pour nous aider à faire la différence entre les licences open source et/ou libres, nous nous sommes basés sur la liste officielle des licences certifiées open source[17] et sur la liste des licences logiciel libre définies dans l'article *Various Licenses and Comments about Them*[18].

Les licences logiciel libre et open source

Nous allons nous baser sur [Laurent, 2004] pour présenter les licences les plus utilisées dans le monde libre. Ces licences sont la MIT, la BSD, l'Apache, la GPL, l'OSL et la MPL. Le texte intégral de ces licences peut-être trouvé dans l'annexe ou sur les sites Internet référencés dans la bibliographie.

Le Copyleft

Le copyleft est un mécanisme original mis au point par la FSF lors de la création de la première version de la GPL. Il consiste à obliger le licencié qui redistribue un programme open source, modifié ou non, à le redistribuer sous une licence open source, la même ou une différente en fonction de la force du copyleft employé dans la licence de base.

Il a donc comme objectif d'empêcher un licencié de licencier un travail dérivé sous une licence propriétaire.

Dans le copyleft, nous pouvons distinguer différentes puissances :

- le copyleft fort (GPL) : tout logiciel qui fait appel à du code sous cette licence doit être mis sous cette licence. Par exemple, lorsqu'une librairie est sous GPL, tout logiciel qui fait appel à celle-ci doit être licencié sous GPL.
- le copyleft normal (OSL) : le logiciel, ses modifications et tout ce qu'on y ajoute doivent être mis sous cette licence.
- le copyleft faible (MPL) : seul le code d'origine et ses modifications doivent rester sous cette licence. Les ajouts peuvent être mis sous n'importe quelle licence, même propriétaire.

Text 1: Le Copyleft

17 Liste des licences certifiées OSI disponible sur le site Internet suivant : http://www.opensource.org/licenses/index.php.

18 Cet article est disponible à l'adresse suivante : http://www.fsf.org/licensing/licenses/index_html.

Licence

Licences non copyleftées

MIT : The MIT license

La licence MIT est la plus simple des licence open source (4 paragraphes). Malgré cela, elle reprend les points suivants :

- la notice de Copyright qui reprend l'année de création et le nom de l'auteur du logiciel,
- les droits d'utiliser, copier, modifier et distribuer le logiciel sans aucune restriction,
- elle autorise l'exploitation commerciale du logiciel ou de ses dérivés mais le nom de l'auteur d'origine doit être inscrit dans les licences,
- elle reprend aussi la clause de non-garantie.

BSD : the BSD License

La licence BSD est un peu plus longue et plus restrictive que la licence MIT,et existe sous différentes formes similaires.

Elle reprend les points suivants :

- la notice de Copyright qui reprend l'année de création et le nom de l'auteur du logiciel,
- elle autorise l'utilisation, la copie, la modification et la redistribution du logiciel sous certaines conditions :
    - la redistribution sous code source doit contenir la « Copyright notice », cette liste de conditions et les clauses suivantes,
    - la redistribution sous forme binaire doit contenir la « Copyright notice », cette liste de conditions et les clauses suivantes dans la documentation et/ou dans les autres documents fournis avec la distribution,
    - on ne peut utiliser le nom de l'auteur original dans le cadre de la promotion d'un logiciel dérivé sans son autorisation préalable,
- elle reprend aussi la clause de non-garantie.

Apache : The Apache License, v2.0

Cette licence est plus complexe et beaucoup plus longue que les deux précédentes. Elle est composée de neuf points distincts :

- Définitions : des termes qui seront utilisés dans les points suivants (ex : licence, licencié, ..etc).
- Accorde entre autre le droit de reproduire, redistribuer le travail et les travaux dérivés.
- Accorde la licence de brevet : le donneur de licence promet aux utilisateurs du logiciel de ne pas les attaquer sur base de ses brevets.
- Redistribution :

    Nous pouvons reproduire et distribuer des copies du travail ou d'un travail dérivé sous n'importe quel forme tant qu'on respecte certaines conditions :

    - on doit fournir une copie de la licence,

    - on doit informer l'utilisateur sur les fichiers qui ont été modifiés,
    - on doit reproduire la copyright notice, la marque et autres informations présentes dans la licence d'origine.

  Il autorise l'utilisation d'une autre licence pour les travaux dérivés.
- Soumission de contributions : tout contributeur au code d'un travail sous MPL doit accepter que sa contribution soit licenciée sous MPL. Ici, on distingue la contribution au travail principal et l'ajout de plug-ins qui peuvent être licenciés sous une licencie autre que la MPL.
- Marques : il est interdit d'utiliser le nom du licenciant pour promouvoir un travail dérivé sans son accord.
- Clause de non-garantie.
- Limitation de responsabilité : du licenciant quant aux problèmes que pourrait causer l'utilisation du travail.
- Accepter des garanties ou des responsabilités supplémentaires : autorise des tiers à proposer des garanties et de prendre des responsabilités complémentaires à celles présentes dans la licence sans mettre en cause celles du licenciant.

Licences copyleftées

Les deux licences qui suivent sont des licences copyleftés ce qui les différencient d'emblée par rapport à celles présentées ci-dessus. Malgré cette différence importante, elles reprennent les idées principales présentes dans ces dernières. Nous allons donc nous attarder sur les points qui les différencient et non décrire chaque clause de ces licences.

GPL : GNU General Public License

Cette licence est proposée par la FSF et est une des plus utilisées par les développeurs de logiciels open source / libre.

C'est une licence copyleftée de type fort.

Quelques spécifications de cette licence :

- On doit toujours indiquer ce que l'on a modifié dans un fichier.
- On doit toujours redistribuer un travail dérivé sous la même licence (ici la GPL).
- On peut ajouter une restriction géographique au travail si c'est nécessaire (brevet sur un objet utilisé dans le travail, etc.)
- On peut décider que le licencié :
    - doit utiliser une certaine version de la licence,
    - peut choisir entre la version indiquée et les suivantes,
    - peut choisir entre toutes les versions de la GPL, passée, présente et à venir.

  Et cela tant que la licence respecte la même philosophie que la licence officielle au moment de la création du travail.

Nous constatons donc que cette licence en plus d'avoir les aspects habituels d'une licence open source (liberté d'utilisation, copie, modification et redistribution) propose des contraintes complémentaires (copyleft fort) et des originalités (choix de la version de la licence).

OSL : Open Software License

La licence OSL est fort semblable à la licence GPL. Elles se distinguent principalement par les points suivants :

- elle n'a aucun article semblable à l'article neuf de la GPL concernant le choix de la version de la licence par le licencié,
- elle fait référence au copyleft normal et pas au copyleft fort comme la GPL. Dans ce cas, nous ne devons pas licencier sous OSL les travaux qui utiliseraient une librairie sous OSL.

Pour voir les autres petites différences entre la GPL et l'OSL, nous conseillons de les lire.

MPL : The Mozilla Public License 1.1

Cette licence est la plus complexe des licences présentées ici. Cela est du au fait qu'elle a été mise au point avec l'aide juridique de Netscape. C'est pourquoi nous n'allons pas voir dans les détails ses différentes spécifications mais que nous allons nous attarder sur sa principale originalité et sa cause.

Sa principale originalité est qu'elle fait appel à ce qui a été appelé plus haut le « copyleft faible ». Cette licence autorise donc toute personne à utiliser du code d'un travail licencié sous cette licence dans un autre travail licencié sous une autre licence ou à ajouter du code non MPL à un travail sous MPL.

La raison de l'utilisation de ce copyleft faible se trouve dans les origines de la création de la MPL. Elle a été mise au point sur base de la Netscape Licence qui est la licence utilisée lors de l'ouverture du logiciel Netscape. Ce logiciel comprenant des plug-ins propriétaires, il était hors de question d'utiliser une licence comme la GPL qui aurait obligé les auteurs de plug-ins à les ouvrir. La solution du copyleft faible fut alors employée pour permettre aux éditeurs de plug-ins propriétaires de continuer à les proposer aux utilisateurs de Netscape et maintenant de Mozilla.

L'utilisation de cette licence n'est évidemment pas limitée à la fondation Mozilla, elle peut être utilisée par tout le monde.

Licence open source et non libre

Pour prouver que, comme montré dans la Figure 3 du premier point de ce chapitre , les logiciels libres peuvent être vus comme un sous-ensemble des logiciels open source, il nous suffit de trouver une licence qui est validée comme open source mais pas comme logiciel libre. C'est le cas des logiciels déposés sous la licence RPL (Reciprocal Public License).

Comme elle est reprise dans la liste des licences certifiées open source, nous pouvons dire que cette licence définit des logiciels open source.

Par contre, l'article *Various Licenses and Comments about Them* cité ci-dessus, comme référence de l'appartenance d'une licence à la catégorie des licences de licences libres, la considère comme une « non-free license » c'est-à-dire une licence non libre. Il en donne trois

raisons :

1. La RPL pose une limite sur le montant de vente d'une première copie,
2. Elle requiert un avertissement au développeur initial pour la publication d'une version modifiée,
3. Elle requiert la publication de toute version modifiée qu'une organisation utilise même si c'est de manière privée.

De par cet exemple, nous pouvons confirmer que les logiciels libres sont des logiciels open source mais que tous les logiciels open source ne sont pas des logiciels libres.

La Shared Source Initiative

Est une initiative de Microsoft. Elle a pour but de proposer différentes licences qui permettent l'accès au code source des produits Microsoft.

Cette initiative rappelle l'Open Source Initiative ce qui n'est probablement pas innocent de la part de Microsoft.

Elle propose donc différentes licences qui donnent au licencié :

- le droit de voir le code source (code arrangé par les employés de Microsoft...)
- le droit de modifier ce code
- le droit de redistribuer le code modifié avec certaines restrictions (principalement dans un but scolaire)

Ces droits sont soit séparés soit pris séparément.

Voici trois licences sorties de cette initiatives qui méritent notre attention :

- Microsoft Permissive Licence (Ms-PL) : est une licence qui ressemble à la BSD,
- Microsoft Community Licence (Ms-CL) est une licence qui ressemble à la MPL et qui est fort proche de la GPL,
- Microsoft Reference Licence (Ms-RL) : est une licence où la modification et la redistribution du code source sont interdites. Elle n'autorise donc que la vue du code.

Plus d'informations sur cette initiative peuvent êtres trouvées sur :

- le site principal de la SSI, http://www.microsoft.com/resources/sharedsource/default.mspx,
- la page présentant plus en détail les trois licences citées ci-dessus, http://www.microsoft.com/resources/sharedsource/licensingbasics/sharedsourcelicenses.mspx.

Text 2: La Shared Source Initiative de Microsoft

4 Les formats de fichiers (Formats ouverts)[19]

Un fichier est un ensemble de données organisées suivant des règles qu'on appelle format de données. Et quand ces données sont enregistrées dans un fichier, nous pouvons parler de format de fichier.

Les fichiers et donc leur format permettent l'échange facile de données entre différentes applications et/ou différentes personnes.

Il existe une quantité innombrable de formats de fichiers, ils sont identifiables par l'extension des fichiers. Voici quelques exemples de formats de fichiers parmi les plus communs :

- Page : DOC, RTF, PDF, PostScript, OpenDocument
- Document : PDF, HTML
- Exécutable : EXE
- Images : JPEG, GIF, PNG, BMP
- Son : MP3, WAV, WMA, AAC, Ogg
- Vidéo : MPEG, AVI

Chacun de ces formats peut être classé en fonction de son appartenance aux principales catégories de formats de fichiers suivantes : ouverts, fermés, propriétaires, standards.

4.1 Les formats ouverts

Un format ouvert est un format de fichier dont l'organisation des données et les spécifications techniques quant à sa création, sa lecture et ses modifications sont libres d'accès. Et ce sans aucune restriction juridique.

L'interopérabilité liée à ces formats et le fait de leur accessibilité sur le long terme sont fortement appréciés par les administrations qui voient dans les formats ouverts une opportunité pour stocker sur le long terme des données sensibles. Mais nous reviendrons sur ce point dans le chapitre 3.

Dans cette catégorie, nous retrouvons entre autre les formats suivants : OpenDocument, RTF, PDF, HTML, JPEG, PNG et Ogg.

4.2 Les formats fermés

Les formats fermés à l'opposé des formats ouverts ont leur spécifications techniques cachées et protégées.

L'utilisation d'un format fermé est souvent utilisés pour capturer le client. En effet, si un fichier sous un format fermé ne peut être créé, lu et modifié qu'avec un seul logiciel, son utilisateur aura tendance à continuer à utiliser les logiciels lisant ce format de fichier et donc ne pas passer à la concurrence.

Il arrive parfois qu'un format fermé soit lisible par un concurrent mais cela demande beaucoup de travail ou coûte très cher. Par exemple, OpenOffice arrive à gérer les documents de Microsoft Office mais cela ne fut réalisable que par du reverse engeniring, ce qui demande

19 Les informations reprisent dans ce point sont la compilation des données trouvées sur les sites Internet suivants : http://fr.wikipedia.org/ et http://www.belgif.be/.

beaucoup de temps et ne donne pas toujours un résultat parfait.

Dans cette catégorie, nous retrouvons entre autre les formats suivants : DOC et GIF.

4.3 Les formats propriétaires

Un format propriétaire est un format créé par une organisation qui en détient donc les droits. Il est généralement créé dans un but commercial. Différents buts peuvent être recherchés pour autant qu'on rende ce format ouvert ou fermé.

Exemples :

- le format PDF est un format propriétaire ouvert donc le but de son éditeur est de rendre son format et ses logiciels incontournable de part leur qualité et non pas de leur « unicité ».
- le format DOC de Microsoft est un format propriétaire fermé donc le but de Microsoft est de fermer le marché à la concurrence et d'obliger les utilisateurs de ce format à continuer à acheter ses logiciels.

4.4 Les formats standards

Les formats standards sont des formats utilisés par un grand nombre de personnes. C'est une définition assez simpliste qui correspond bien avec la traduction anglaise du mot « standard » qui signifie *norme* en français. Par exemple, le format DOC est considéré comme un standard de fait de son omniprésence dans le monde informatique.

Il existe aussi des organismes qui définissent des standards comme le W3C qui définit les standards du langage HTML. Ce n'est donc pas toujours le fait de sociétés privées.

Les formats en phase de standardisation et en vogue actuellement sont les formats basés sur le XML comme par exemple l'OpenDocument utilisé par OpenOffice.

Les logiciels open source travaillent généralement avec les formats standards ouverts mais ils peuvent dans certains cas proposer des implémentations des formats fermés avec quelques limites évidemment (exemple : OpenOffice qui gère les formats de Microsoft Office).

Les principes d'un standard d'après *Gordon* Bell (Microsoft) et Rob Gingell (Sun Microsystems)

Un standard :

1. Précise des points d'homogénéité, autorisant toute variété et toute innovation sur les points non spécifiés.
2. Se fond par principe sur une *spécification*, pas une *implémentation*.
3. Importe plus pour ce qu'il apporte à son utilisateur que par ses qualités propres.
4. Concerne et n'est censé affecter que la communauté qui y adhère.
5. Ne vaut que par l'efficacité de l'organisme qui confère les certifications à ce standard.
6. Ne doit pas, pour être adopté, « faire table rase du passé »
7. Doit laisser une place adéquate aux innovations futures
8. Disparaît quand il devient un obstacle à l'innovation au lieu d'en être une plateforme.

Text 3: Les principes d'un standard d'après Gordon Bell et Rob Gingell

Chapitre 2 : Les personnes qui font vivre l'Open Source

Ce chapitre va nous emmener dans le monde des personnes oeuvrant pour l'Open Source. C'est pourquoi, le premier point de ce chapitre sera consacré aux personnes qui influencent le monde de l'Open Source et du libre.

Le deuxième point parlera lui des communautés Open Source. Nous présenterons leur mode organisationnel, les différents types de communautés qui existent, les motivations qui poussent des personnes à les rejoindre et la principale méthode de développement de ces communautés.

1 Les acteurs du monde libre

Dans cette partie, nous allons nous attacher à la présentation de certains acteurs importants de la communauté libre. Ce sont soit des personnes physiques soit des organisations. Nous avons classé ces acteurs en fonction de leur appartenance principale à l'une ou l'autre mouvance : Logiciel Libre ou Open Source.

La plupart des informations données sur les acteurs du monde libre ont été trouvées sur des sites Internet et les sites officiels des différentes organisations définies ci-dessous.

1.1 Les principaux acteurs de la mouvance Logiciel Libre

1.1.1 Richard M. Stallman

Il est né en 1953, fait des études universitaires à Harvard et en sort en 1974 avec un diplôme en physique. Il a travaillé jusqu'en 1985 au « MIT Artificial Intelligence Lab ». Ensuite il fonde le projet GNU en 1986. Il est d'ailleurs très connu pour la multitude de logiciels qu'il a développés, comme par exemple « Emacs » et « GCC ». Il travaille toujours dans les logiciels libres en tant que membre de la FSF qu'il a fondé en 1985. Son pseudonyme, « RMS », est très connu dans le monde des hackers.

1.1.2 Free Software Foundation (FSF)[20]

La FSF a été créée en 1985 par Richard Stallman et d'autres participants au projet GNU. Son rôle est de promouvoir le développement et l'utilisation des logiciels libres par des publications, des conférences, et tout autre moyen qu'elle juge nécessaire. Elle défend, aussi, le droit des utilisateurs à employer, étudier, copier, modifier et redistribuer les logiciels. Elle vit grâce à des donations, la vente de logiciels libres et celle de livres. Elle est aussi à la base de la GPL (General Public License), qui est la licence logiciel la plus utilisée par les logiciels open source, et de la LGPL (GNU Library or « Lesser » General Public License)[21].

La devise de la FSF est « *Free software is a matter of liberty not price. You should think of "free" as in "free speech"* ».

1.1.3 GNU Project[22]

Le projet GNU a été lancé en 1984 par Richard Stallman qui voulait d'après ses mots « *ramener l'esprit de coopération qui prévalait dans la communauté informatique dans les jours anciens* ».

Ce projet a pour objectif de développer un système d'exploitation complet et libre. Ce système doit ressembler à un système UNIX du point de vue de la stabilité, la « facilité » de gestion et de la portabilité. GNU est l'acronyme de GNU's Not Unix, ce qui prouve le souhait des initiateurs du projet de développer un UNIX qui n'est pas UNIX.

Grâce à l'arrivée de Linux, en 1991, ce projet fut enfin viable. D'ailleurs on parle souvent de GNU/Linux car le noyau s'appelle « Linux » mais les autres parties du système viennent du projet GNU. Le projet GNU développe aussi son propre noyau baptisé « Hurd ». Ce noyau n'est à l'heure actuelle pas « commercialisable ».

Le projet GNU est principalement sponsorisé par la FSF.

1.2 Les principaux acteurs de la mouvance Open Source

1.2.1 Bruce Perens

Il a travaillé pendant de nombreuses années dans l'industrie de l'animation graphique dont 12 années (1987-1999) pour les studios *Pixar*. Il a aussi travaillé chez *Hewlett-Packard Corporation*. Et travaille maintenant chez *Perens LLC* qui a, notamment, comme clients IBM, NTT (Compagnie japonaise de télécommunication), Philips, Novell et Borland.

Il est plus connu en tant que figure importante du mouvement open source. En effet, il a été chef du projet Debian[23]. Premier auteur de la définition de « Open Source », il est un des co-fondateurs de l'Open Source Initiative.

20 Pour plus d'informations sur la FSF, vous pouvez consulter son site Internet : http://www.fsf.org ou son pendant francophone : http://www.fsffrance.org.

21 La LGPL est une licence fort proche de la GPL qui s'applique principalement aux librairies. De plus amples informations sur la LGPL peuvent être trouvées sur : http://fr.wikipedia.org/wiki/LGPL.

22 Pour plus d'informations sur le projet GNU, vous pouvez aller visiter son site Internet : http://www.gnu.org/.

23 Le projet Debian est à la base de la distribution Linux portant le même nom. Pour plus d'informations, visitez le site Internet suivant : http://www.debian.org/.

1.2.2 Eric S. Raymond

Il est né en 1957 et a voyagé à travers tous les continents avant de se fixer en Pennsylvanie. Il est bien connu du monde open source. En effet, il a longtemps collaboré au développement de plusieurs logiciels libres avant de devenir un défenseur publique de l'open source. Il est d'ailleurs un des co-fondateurs de l'Open Source Initiative. C'est aussi une personne publique assez controversée à cause de ses idées tranchées sur les armes, la guerre en Irak, etc. Il connaît bien le milieu des hackers ce qui lui a permis d'écrire *The Cathedral and the Bazaar* et *The New Hacker's Dictionary*. Deux ouvrages qui ont eu un énorme succès et sont considérés par certains comme des bibles.

1.2.3 Tim O'Reilly

Il est né en 1954 à Cork en Irlande et est sorti en 1975 de Harvard avec un diplôme en Classique.

C'est le fondateur et CEO d' *O'Reilly Media* qui est connu comme étant un éditeur de livres informatiques parmi les meilleurs. C'est un ardent défenseur de la cause open source grâce à son rôle d'éditeur.

1.2.4 Open Source Initiative (OSI)[24]

L'OSI a été fondée en 1998 par Eric S Raymond et Bruce Perens. C'est une « ASBL » (non-profit corporation). Son but est de promouvoir la définition de l'open source et tout ce qui en découle, notamment, les logiciels open source et les licences open source. Elle décerne le label OSI à tout logiciel qui respecte, via sa licence, la définition de l'open source.

Elle propose sur son site Internet la liste des licences qui sont acceptées comme respectant la définition de l'open source.

1.3 Les acteurs non classables dans l'une ou l'autre mouvance

1.3.1 Linus Torvalds

Linus Benedict Torvalds est né en 1969 à Helsinki, en Finlande. Il a étudié à l'université d'Helsinki. Lors de ses études, il a commencé à développer, durant son temps libre, Linux qui est inspiré de Mimix, un noyau d'étude. Il est actuellement le coordinateur du développement de Linux et travaille à l'Open Source Development Lab (OSDL) en Californie.

1.3.2 Berkeley, Université de Californie[25]

Berkeley est la base de la distribution BSD (Berkeley Software Distribution) et est dérivée de UNIX. Elle débuta dans les années 1970. Berkeley arrêta la distribution de BSD en 1995 avec la sortie de la version 4.4BSD-Lite Release 2.

24 Pour plus d'informations sur l'OSI, vous pouvez aller visiter son site Internet : http://www.opensource.org/.

25 Pour plus d'informations sur l'histoire du projet BSD, vous pouvez aller visiter le site Internet http://en.wikipedia.org/wiki/Berkeley_Software_Distribution, et les liens qu'il propose.

La version 4.4BSD-Lite servit de base pour de nouvelles distributions comme FreeBSD, OpenBSD, et NetBSD. La licence de BSD permit à d'autres systèmes d'exploitation tant libres que propriétaires d'incorporer des parties de son code source. Ainsi Microsoft a utilisé du code dérivé de BSD dans son implémentation de TCP/IP.

2 Les communautés Open Source

Maintenant que nous savons ce qu'est un logiciel open source, il est nécessaire de comprendre comment de tels logiciels émergent, comment et par qui ils sont gérés. Ce qui va nous amener à parler des communautés open source.

Les premières questions à se poser sont : qu'est ce qu'une communauté open source, comment est-elle gérée et pourquoi rejoindre une communauté open source.

Ensuite, nous aborderons la question du mode de développement d'un projet open source en comparaison au mode de développement des logiciels propriétaires.

Finalement nous nous attarderons sur les moyens mis à disposition de ces communautés, leur visibilité et leur évolution en fonction de leur maturité.

2.1 Les communautés

Pour faire simple, une communauté open source est un groupe de personnes ayant le même objectif. Cet objectif est de participer à un projet open source. L'objectif d'un projet open source est, en général, de développer un ou plusieurs logiciels open source. Il existe, bien évidemment, des projets open source qui ont d'autres objectifs comme par exemple : la mise en place de documentations, la création de distribution (projet Debian, Ubuntu ...), etc.

La plupart des projets open source sont portés par un nombre restreint de personnes, souvent de une à cinq dans ses débuts. Il existe, aussi, des projets open source de plus grande ampleur comme les projets Mozilla, OpenOffice, Apache qui ont un nombre important de participants.

Dans les petites communautés, la structure est généralement assez simple, il y a un chef de projet, souvent l'initiateur du projet, qui a tous les droits. Il décide ce qui peut ou pas être ajouté au code, si un nouveau participant peut avoir des droits d'accès privilégiés au code du projet, etc. Le chef est donc omnipotent.

Dans les communautés qui prennent une certaine importance, il n'est pas toujours raisonnable de rester avec une structure simple. Il y a donc une évolution de la structure avec la création de différents postes, de différents département, de différents niveaux de pouvoirs. Nous allons décrire les postes qui reviennent le plus fréquemment. Ils sont la compilation de différentes sources : [Raymond, 2001], [Koch, 2005] et d'autres documents référencés dans la bibliographie.

Une communauté open source mature est souvent constituée de la manière suivante :

- **Un gestionnaire de projet** : il est généralement l'initiateur du projet ou une personne qui y a apporté une contribution fondamentale. Il gère tout ce qui concerne le développement du projet. C'est lui qui décide si la proposition d'un membre doit être ajoutée dans le logiciel, ou pas.

- **Un comité de gestion** : il peut parfois se substituer à un gestionnaire de projet. Il en existe deux sortes :
  - les comités élus par l'ensemble des membres de la communauté, donc le choix des leaders se fait de manière démocratique,
  - les comités auto-proclamés, c'est-à-dire que le comité en place décide qui peut y entrer soit pour remplacer quelqu'un soit pour augmenter la taille du comité.
- **Des développeurs actifs** : ils sont peu nombreux, sont respectés par la communauté et ont la confiance du directoire. Ils ont la possibilité de poster directement des modifications dans le CVS[26] du projet. C'est dans leur rang que sont généralement choisis les nouveaux membres du comité de gestion. Une grande partie de la vie d'une communauté repose sur eux. Ce sont eux qui parrainent des développeurs moins expérimentés.
- **Des développeurs périphériques** : ils sont très nombreux mais leur contribution est souvent minime. En effet, ils ne participent qu'une ou deux fois au projet soit pour le comprendre, soit pour apporter des modifications dont ils ont besoin dans leur activité quotidienne. C'est pourquoi dans ce groupe, on retrouve principalement des informaticiens qui utilisent ou gèrent ce logiciel dans leur activité professionnelle.
- **Des Bug Reporter** : ce sont des utilisateurs réguliers du programme. Ils rapportent à la communauté les problèmes qu'ils rencontrent. Ils demandent aussi des changements pour améliorer l'efficacité du programme en fonction de leurs besoins spécifiques. Ils sont donc très importants pour la communauté.
- **Des lecteurs** : ils sont très rares. C'est généralement le premier stade avant de devenir un développeur périphérique puis un développeur actif si leurs contributions sont intéressantes et utiles. Comme le nom l'indique, ils lisent le code source et peuvent proposer des modifications de style ou des corrections s'ils trouvent des « erreurs » de programmation. La plupart d'entre eux font cela soit pour comprendre le fonctionnement interne du programme, soit pour apprendre le langage de programmation utilisé par la communauté. En effet, le code source d'un logiciel libre est, généralement, lisible et bien documenté. Car les programmeurs savent que leur code sera lu par d'autres et ils préservent leur réputation de bon programmeur en rendant ce code accessible au plus grand nombre.
- **Des utilisateurs passifs** : ils représentent la majorité des « membres » d'une communauté open source. Ce sont, d'ailleurs, les membres les plus importants. Car un logiciel inutilisé n'a aucun intérêt à être développé.

Il est évident que chaque communauté a ses spécificités, donc des postes qui ne sont pas repris ci-dessus.

Nous constatons donc qu'il est parfois nécessaire de faire évoluer la structure organisationnelle d'une communauté en même temps que le projet prend de l'ampleur ou change d'état. Pour bien comprendre ce fait, utilisons une analogie au monde économique. Dans le développement d'un produit, il y a d'abord le stade de recherche et développement.

26 CVS, *Concurrent Versions System*, est un logiciel open source de gestion de version. Pour plus d'informations, vous pouvez visiter le site Internet suivant : http://www.nongnu.org/cvs/.

Puis une fois le produit prêt, il passe au stade de la commercialisation. Ce changement de stade est alors accompagné d'une modification organisationnelle.

Pour un logiciel, c'est la même chose. Si l'organisation n'évolue pas assez vite, le risque est grand de voir apparaître des schismes du projet initial.

2.2 Différents types de communautés

Nous allons essayer de déterminer différents types de communautés. Ces types de communautés sont déterminés en fonction du support logistique dont elles disposent : les moyens personnels, les moyens d'une société, les moyens d'une fondation, etc.

Ce n'est pas parce que deux communautés appartiennent à des types différents qu'elles ne peuvent pas avoir le même genre d'organisation.

Nous avons déterminés trois types principaux :

2.2.1 Les communautés de « base »

Ce sont des communautés de fait. A la base, il y a une personne ou un groupe de personnes qui a décidé de développer un logiciel en fonction de leurs besoins personnels. Ces communautés sont indépendantes de tout support extérieur. On n'y retrouve que des passionnés par le logiciel qu'ils développent. La majorités des communautés connues appartiennent à cette catégorie. Elles n'ont généralement qu'un projet développé par un tout petit nombre de personnes.

2.2.2 Les communautés « sponsorisées »

Ces communautés ont généralement atteint un seuil critique. Elles sont sponsorisées et soutenues par une fondation ou une ASBL. Grâce à ce soutient externe, elles peuvent faire de la publicité, elles ont des développeurs professionnels. Et sont très structurées. (le projet GNU, soutenu par la FSF).

2.2.3 Les communautés « issues du privé »

Ces communautés sont basées sur un logiciel propriétaire qui a été relâché sous licence open source. Elles dépendent, généralement, encore de la société éditrice du logiciel.

Cette catégorie regroupe aussi les communautés dirigées par une société qui développe des logiciels open source.

Chaque projet peut à tout moment changer de catégorie. Et il n'est pas toujours évident de classer un projet dans tel ou telle catégorie. Par exemple, le projet Mozilla a débuté sa vie en tant que communauté privatisée mais elle peut maintenant prétendre être classée dans la catégorie des communautés sponsorisées. Et cela, bien qu'elle ait encore des liens étroits avec son éditeur d'origine. Pour mieux comprendre le cas de Mozilla nous conseillons la lecture du premier chapitre de [DiBona, Cooper and Stone, 2005].

Il y a aussi de plus en plus de sociétés qui se basent sur l'open source pour améliorer leur code ou l'utiliser dans des logiciels propriétaire. Cette nouvelle tendance est appelée le « Source Coding ».

2.3 Pourquoi rejoindre une communauté Open Source ?

Comme expliqué dans [CSC, 2004], ce n'est pas l'argent qui pousse les développeurs des communautés. En effet, il existe des bénévoles partout, pourquoi pas dans le développement de logiciels? D'autant plus, que l'histoire nous a montré qu'à ses débuts, l'informatique était basée sur une communauté d'échanges. Voyons maintenant quelles sont alors les motivations principales de ces bénévoles de l'informatique.

D'après [Viseur] et la Figure 7 reprise du « Boston Consulting Group », nous pouvons dire que les raisons principales pour un hacker de participer à une communauté open source sont :

- la stimulation intellectuelle,
- l'amélioration de ses capacités.

Et non pas, comme certains pourraient le croire, l'obsession de combattre les logiciels propriétaires et d'imposer les logiciels open source.

[Viseur] propose de classer les hackers en quatre catégories : les croyants, les professionnels, les hédonistes et les experts. Voyons quels sont les objectifs principaux de chacun de ces groupes.

Les croyants

Ce sont des personnes poussées par la conviction « quasi religieuse » que le code source des logiciels devrait être ouvert. Cette motivation se situant en troisième position, nous pouvons dire qu'il n'y a pas que les représentants de cette catégorie qui ont cette motivation même si, dans certains cas, ce n'est pas la motivation principale. Leur objectif est donc de prôner l'ouverture du code source. De plus, ce sont de farouches combattants des logiciels propriétaires.

Nous retrouvons donc, d'après leur motivation première, les principaux défenseurs de la philosophie prônée par les logiciels libres et l'open source.

Les professionnels

Cette catégorie regroupe des personnes ayant différentes motivations plus axées sur leur métier et leurs besoins professionnels. Nous retrouvons donc comme motivations principales : les besoins du métier et le statut professionnel.

Mais nous retrouvons, aussi, dans cette catégorie les gens obligés d'utiliser les logiciels open source. Cette « motivation » se situe tout de même en sixième position avec plus de 28%. Cet état de fait nous montre que l'utilisation des logiciels open source, de manière professionnelle, se développe. Malheureusement, il n'est pas expliqué si cette obligation était bien perçue ou pas par les développeurs.

Les hédonistes

Nous retrouvons dans cette catégorie, les personnes qui participent au développement de logiciels open source pour leur « plaisir ». En effet, leurs motivations principales sont les besoins privés et la stimulation intellectuelle. Les besoins privés peuvent être assimilés dans le graphique à différentes motivations : « non-work functionnality », « work functionnality », « work with team », « open source reputation » et « other »... Nous voyons donc que les besoins privés ne sont pas très bien définis et qu'ils peuvent prendre différents aspects en fonction de chacun.

Notons toutefois que la recherche de stimulation intellectuelle est la première motivation qui

ressort de cette étude. C'est aussi la motivation la plus citée, avec l'amélioration des capacités, dans la littérature sur le sujet. En effet, développer un logiciel open source demande un travail intellectuel important.

Les experts

Ce sont principalement des professionnels de l'informatique qui ont trouvé un moyen complémentaire d'améliorer leurs compétences. Leur principale motivation est d'ailleurs d'améliorer leurs capacités. En regardant plus loin, nous pouvons supposer qu'améliorer leurs capacités n'est probablement pas leur seule motivation. En effet, ils sont sûrement aussi motivés par la stimulation intellectuelle, le fait de faire partie d'une communauté de développement et d'être reconnus par leur paires en tant qu'experts.

Cette motivation, l'amélioration de capacités, n'est pas limitée à ce groupe de personnes. En effet, plus on pratique, plus on progresse. Donc un développeur débutant peut avoir comme motivation d'améliorer ses capacités en participant au développement de logiciels open source.

Nous voyons que chaque catégorie a une motivation principale. Mais rien n'empêche quelqu'un appartenant à une catégorie d'avoir des motivations secondaires que l'on retrouve comme motivation principale d'un autre groupe.

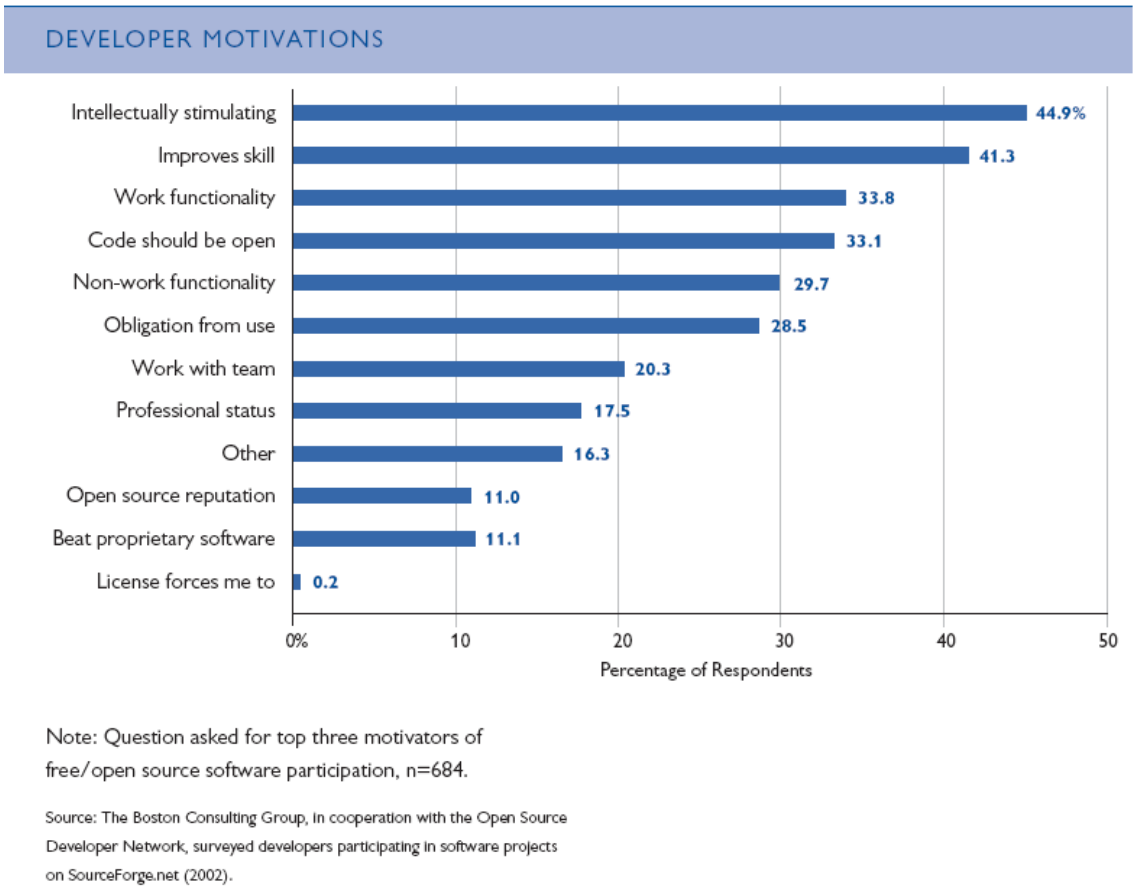


Figure 7: La motivation des développeurs open source selon le Boston Consulting Group, 2002.

De manière plus pragmatique, nous pouvons constater que la majorité des personnes qui participent à une communauté open source le font, principalement, parce qu'ils utilisent le logiciel de cette communauté soit à des fins personnelles, soit à des fins professionnelles. C'est pour cette raison que les logiciels open source sont, au moins, de plus en plus souvent équivalents à leurs homologues propriétaires. En effet, ces personnes désirent utiliser un logiciel performant qui répond à leurs besoins. Si ce n'est pas le cas, ils ont alors la possibilité de le modifier à leur convenance.

2.4 Mode de développement des logiciels open source

The Cathedral and The Bazar, [Raymond, 2001], classe les méthodes de développement en deux catégories principales :

- la méthode « Cathédrale », qui est principalement la méthode de développement des logiciels propriétaires,
- la méthode « Bazar », qui est la méthode principale de développement des logiciels open source.

Ces deux méthodes sont deux grandes catégories théoriques. Il est donc fréquent, en pratique, de trouver des méthodes situées entre celles-ci.

Nous allons décrire ces deux méthodes et voir les avantages de l'une et de l'autre.

2.4.1 La méthode « Cathédrale »

La méthode de la cathédrale est utilisée dans le monde industriel par des entreprises comme Microsoft, IBM, etc. Elle consiste à créer un programme comme il nous l'est enseigné lors de nos études d'informatique.

Développer un programme comporte toute une série d'étapes qu'il est impératif de suivre pour avoir un logiciel robuste, respectant les besoins des clients, bien documenté, testé, etc. Si ces étapes ne sont pas suivies dans l'ordre et dans les temps impartis, nous en payons le prix (pour les étudiants : des points ; pour des sociétés : de l'argent). La méthode Waterfall, Figure 8, peut être considérée comme une méthode Cathédrale.

Cela n'empêche pas l'oubli de bugs ou de trous de sécurité comme nous le voyons régulièrement avec les logiciels de Microsoft par exemple.

2.4.2 La méthode « Bazar »

La méthode du bazar est utilisée dans les communautés open source. Elle consiste, dans la majorité des projets[27], à ce que chacun fait ce qu'il veut, et quand il le veut. En fonction de ses capacités et de ce qui « doit » être fait.

Il n'y a généralement pas de temps imparti donc pas de retard. Il existe toutefois, dans les projets matures, une certaine ligne de conduite et un certain planning à suivre. Ce planning contient, généralement, ce qui devra être fait, dans les grandes lignes, pour la prochaine version et pour quand cela devrait être fonctionnel. Le « quand » est souvent une période assez vague. Par exemple : « l*a fonctionnalité X devra être réalisée pour le premier trimestre*

27 La grande majorité des projets open source que l'on peut trouver sont composés de quelques personnes. A la différence des grands projets comme OpenOffice, par exemple, qui a une communauté composée de milliers de personnes.

2007 ou pour fin juin 2006 ».

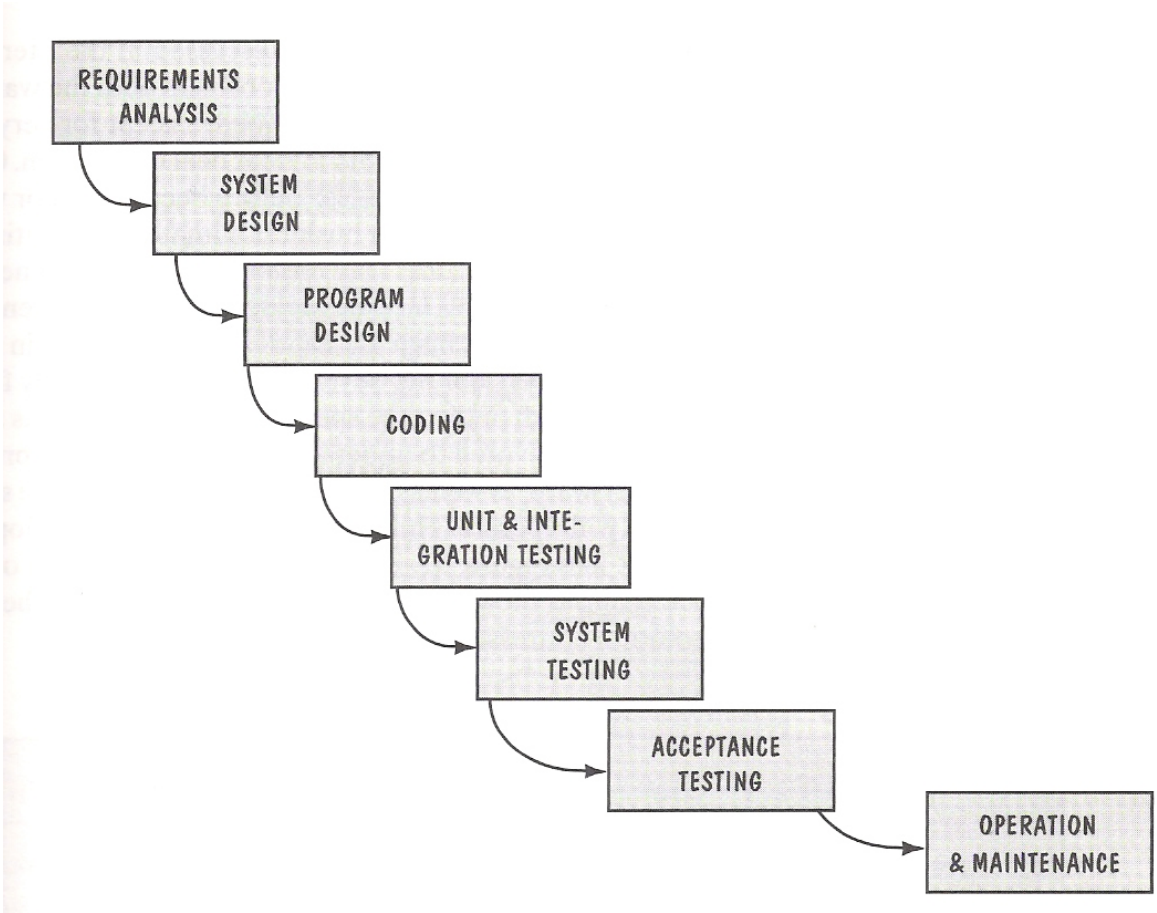


Figure 8: The waterfall model, repris de Shari Lawrence Pfleeger, « Software Engineering theory and practice », New Jersey, Prentice hall, 2001

Si quelqu'un trouve un bug ou un trou de sécurité, soit il propose un patch, s'il en a les capacités, soit il le signale à la communauté qui va se charger de résoudre le problème dans les plus brefs délais. En effet, la dynamique de développement, qu'on pourrait croire lente et peu ordonnée, est rapide et efficace. Contrairement à Microsoft, où il faut généralement un certain temps avant qu'un patch sécurité ne soit proposé.

Comme une communauté est composée de membres qui ont des compétences et des statuts différents, les différentes étapes qu'on retrouve dans le développement habituel d'un logiciel sont réalisées en continu grâces aux releases fréquentes qui peuvent être testées par tous.

Dans le chapitre quatre de [DiBona, Cooper and Stone, 2005], Ben Laurie explique qu'entre le moment où un bug est corrigé et le moment où il est distribué aux utilisateurs par les groupes de distributions (Debian, Suse, ...), le temps écoulé peut être assez long. C'est, principalement, du au fait qu'ils préfèrent analyser ce qui a été fait et le certifier conforme à leurs exigences. Les utilisateurs expérimentés peuvent directement se rendre sur le site officiel du logiciel et récupérer le patch juste après sa mise au point.

Ce mode de développement évolue avec le temps. En effet, dans une communauté qui évolue et met en place une organisation « précise », certaines personnes n'ont plus toujours le « choix » de faire ce qu'ils veulent et comme ils veulent. Tout cela à cause de la mise en place de certaines règles de conduite et de processus d'approbation des développements réalisés.

Mais nous retrouvons toujours cette méthode dans les petits projets qui sont portés par quelques personnes.

2.4.3 Avantages et inconvénients de chaque méthode

[AWT, 2005] nous propose tant des avantages que des inconvénients des logiciels open source et des logiciels propriétaires. Nous pouvons aussi trouver dans la littérature sur les logiciels open source d'autres avantages et inconvénients. Malheureusement, beaucoup sont biaisés par un parti pris. Ici, en nous basant sur [AWT, 2005] nous allons proposer une série d'avantages et d'inconvénients des logiciels open source et des logiciels propriétaires basée sur les deux méthodes de développement présentées ci-dessus, voir la Table 1.

Voici la liste et l'explication des différents points abordés dans la Table 1 :

- Flexibilité : la possibilité d'adapter, via paramétrage, le logiciel à ses besoins.
- Pérennité : durée de vie d'un logiciel depuis sa création jusqu'à son abandon.
- Disponibilité : facilité d'acquérir et de transférer un logiciel.
- Respect des normes : proposées par des organismes « indépendants » et disponibles par tous.
- Performance : du code source. Donc calcul de la qualité du code.
- Fiabilité : du code source, c'est-à-dire un code qui ne contient pas d'erreur ou le moins possible.
- Ergonomie : des interfaces graphiques, de la gestion des menus.
- Homogénéité : entre les produits.
- Maturité : des logiciels.

Nous pouvons conclure ce point sur les communautés open source en comparant une communauté à une société qui évolue en fonction de sa réussite. Il est donc difficile de comparer les communautés entre elles car elles ont chacune leur spécificité propre comme il est difficile de comparer deux sociétés qui ont une structure organisationnelle différente.

	Logiciels open source	***Logiciels propriétaires***
Flexibilité	Le paramétrage des logiciels open source est une de leur plus grande force.	Le paramétrage des logiciels propriétaires est souvent plus limité que celui des logiciels open source.
Pérennité	Le code source étant toujours disponible, son évolution est assurée	La durée de vie d'un logiciel propriétaire est généralement de 3 à 5ans et est dictée par des raisons économiques.
Disponibilité	Un logiciel open source est facilement acquis et peut être tout aussi facilement transféré.	La disponibilité de ce type de logiciel est fortement liée à sa licence et à son mode de distribution.
Respect des normes	Les développeurs ont une forte tendance à respecter les normes internationales car elles sont mises à disposition de tout un chacun.	Pas toujours enclin à utiliser ses normes. Ils utilisent volontiers des normes propriétaires pour cadenasser les clients à leurs produits.
Performance	Comme le code source est ouvert et que tout le monde peut y avoir accès, les développeurs font un effort particulier pour rendre leur code lisible et accessible.	Le code est souvent fermé donc la performance est fortement dépendante des normes imposées aux développeurs par leurs employeurs.
Fiabilité	Débugage et réparation de problèmes facilités par la disponibilité du code source et son accès aisé.	La mise à disposition de patch est souvent plus longue pour ce type de logiciel que pour les logiciels open source.
Ergonomie	Elle laisse généralement à désirer bien que des efforts son faits depuis quelques temps pour prendre en compte ce point important pour les simples utilisateurs.	Elle est souvent placée au premier plan dans cette catégorie de logiciels destinés au grand public.
Homogénéité	Très peu d'homogénéité entre les logiciels open source.	Très forte homogénéité entre les produits d'un même développeur.
Maturité	La maturité dépend de chaque logiciel et est souvent difficile à évaluer.	Ces logiciels sont souvent considérés comme matures du moins pour les plus connus et utilisés d'entre eux.

Table 1: Avantages et Inconvénients des logiciels propriétaires et open source selon notamment [AWT, 2005].

Partie 2

Migration

Chapitre 3 : Enjeux et risques de l'utilisation de l'Open Source

Ce chapitre est divisé en deux points qui ont pour objectif d'abord de présenter les enjeux et avantages d'un passage à l'open source, ensuite de présenter les risques liés à ce passage.

Nous allons donc proposer les principales raisons qui peuvent nous pousser à envisager l'utilisation de solutions basées tout ou en partie sur des logiciels open source et les risques auxquels il faudra faire attention lors de ce passage.

1 Analyse des enjeux stratégiques pour les administrations publiques de passer à l'open source

Il existe différents enjeux qui s'offrent à toute administration qui désirerait passer totalement ou en partie à l'open source. Les principales opportunités sont :

- l'utilisation des standards ouverts
- l'économie faite sur les licences
- l'évolution des logiciels et de l'innovation
- l'augmentation des compétences du personnel
- la possibilité de ne plus dépendre d'un éditeur unique
- etc.

1.1 Les enjeux politiques liés aux formats standards ouverts

Jean Jochmans et Peter Strickx dans un rapport du FedICT[28] définissent les standards ouverts comme suit :

28 Directives et recommandations pour l'usage de standards ouverts et/ou spécifications ouvertes dans les administrations fédérales, du 6/10/2004 (= annonce sur site Internet suivant : http://www.belgium.be/eportal/application?origin=newsFPSList.jsp&event=bea.portal.framework.internal.refresh&pageid=contentPage&docId=36436)

1 **Spécification ouverte**
Une “spécification ouverte” doit être gratuite, disponible en ligne et suffisante pour développer une implémentation complète.

2 **Spécification libre**
Une “spécification libre” doit être ouverte (réf. 1) et ne doit pas comprendre de restrictions juridiques (à l’exception de “licences open source”) qui compliquent la diffusion et l’utilisation.

3 **Standard ouvert**
Un “standard ouvert” est une “spécification libre” (réf. 2) et doit être approuvé par une organisation de standardisation indépendante.

Dans la Figure 9, qui suit, on peut voir clairement la répartition des formats les plus usuels. Dans cette figure, reprise du rapport cité précédemment, il n'était pas encore question du format open document qui est le nouveau standard prôné par une multitude d'administrations de par le monde.

> « *Singapore's Ministry of Defense, France's Ministry of Finance and its Ministry of Economy, Finance, and Industry, Brazil's Ministry of Health, the City of Munich, Germany, UK's Bristol City Council, and the City of Vienna in Austria are all adopting applications that support OpenDocument.* » (OASIS, 2005b)[29].

Ce nouveau format est basé sur le standard ouvert XML. Le format OpenDocument est utilisé nativement par la suite bureautique OpenOffice.org et pas encore par Microsoft Office, ce qui pourrait être préjudiciable à Microsoft qui annonçait, milieu 2005, l'utilisation du XML comme format par défaut pour sa nouvelle version d'Office[30]. Et plus récemment, il a annoncé sa volonté de standardiser le format de document Office Open XML[31].

Le principal avantage de l'utilisation des standards ouverts est la pérennité des données. En effet, au fil des versions de la suite Office de Microsoft, on a pu constater que les formats « doc, xls et ppt » évoluaient sans qu'on ne puisse rien y faire et, par exemple, un document créé sous Office XP n'est plus lisible avec Office 97. Ce qui nous oblige à changer régulièrement de version d'Office et à payer une nouvelle licence si nous voulons continuer à communiquer avec nos collaborateurs utilisant une autre version plus récente d'Office.

Les formats ouverts ne sont pas à l'abri des évolutions mais comme ils sont principalement basés sur le format XML qui est un format connu et bien spécifié, il sera toujours possible de mettre au point des logiciels de « migration en masse » de documents ou d'ajouter des patches qui permettront de prendre en compte les évolutions de ces formats dans d'anciennes versions d'un logiciel spécifique. De plus, comme la plupart des logiciels utilisant les formats ouverts

29 Information trouvée sur le site Internet suivant : http://en.wikipedia.org/wiki/OpenDocument#Public_policy_implications.

30 Microsoft Makes XML the File Format for the Next Version of Microsoft Office, 1/06/2005, http://www.microsoft.com/presspass/features/2005/jun05/06-01XMLFileFormat.mspx.

31 - Q&A: Microsoft Co-Sponsors Submission of Office Open XML Document Formats to Ecma International for Standardization, 21/11/2005, http://www.microsoft.com/presspass/features/2005/nov05/11-21Ecma.mspx - Microsoft Offers Office Document Formats to Ecma International for Open Standardization, 22/11/2005, http://www.microsoft.com/presspass/press/2005/nov05/11-21EcmaPR.mspx.

sont des logiciels open source, donc « gratuits », il sera plus aisé de passer à une version supérieure.

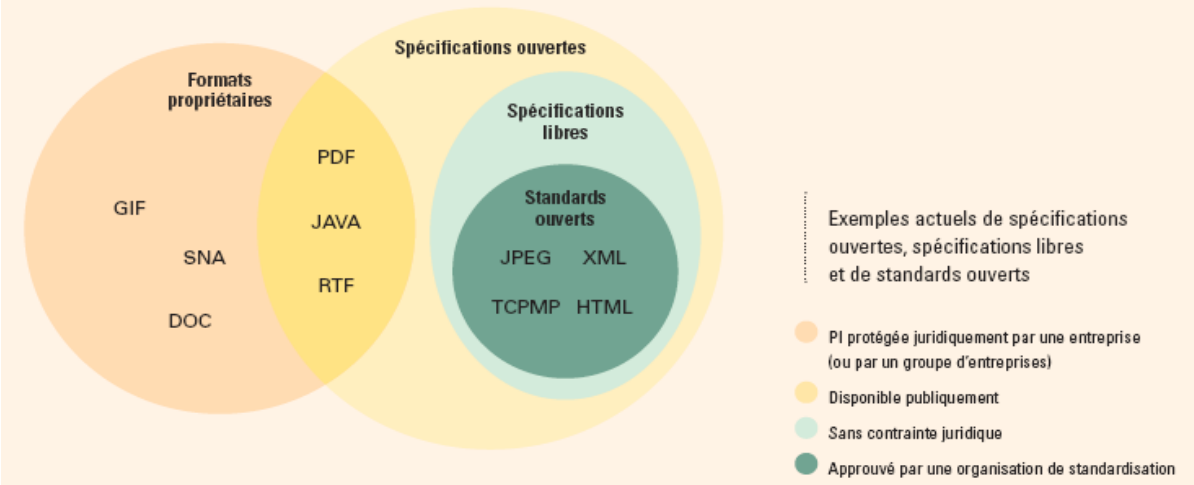


Figure 9:Exemple de formats ouverts et propriétaires

Un autre avantage des formats ouverts est qu'ils peuvent être utilisés, créés et lus par des applications différentes. Par contre, un fichier « doc » ne peut en général être lu correctement qu'avec Microsoft Word. Les formats ouverts sont donc indépendants du système utilisé. Il suffit d'avoir accès à une application permettant de lire ce format. Ces applications peuvent très souvent être téléchargées gratuitement sur internet. Comme par exemple « Adobe Reader[32] » qui permet de lire le format « pdf ».

Pour conclure sur ce sujet, je citerais une phrase de Jan Verlinden de la ville de Schoten[33] :

> *« It is not defendable to use closed source client software like MS Word and to oblige citizens to buy expensive software to be able to read information from the Government ».*

1.2 Les enjeux économiques liés aux coûts des licences

Lorsqu'on parle du coût des licences, il ne faut pas oublier le coût lié à la gestion des licences. L'utilisation des logiciels open source permet généralement de faire des économies sur les coûts des licences (généralement gratuites) mais, comme il existe une soixantaine de licences certifiées open source[34], la gestion de ces licences peut parfois se révéler très difficile et financièrement onéreuse. En effet, il faut toujours vérifier que les termes d'une licence sont compatibles avec l'utilisation que l'on veut faire du travail licencié.

32 Pour plus d'informations, veuillez visiter le site Internet suivant : http://www.adobe.be.
33 Informations sur la migration de la ville de Schoten sur le site Internet suivant : http://europa.eu.int/idabc/en/document/5178/470.
34 Voir le site internet suivant pour voir toutes les licences certifiées open source : http://www.opensource.org/licenses/index.php.

En Belgique

En Belgique, à tous les niveaux de pouvoirs, les choses bougent en faveur de l'utilisation des formats standards ouverts.

Au niveau fédéral, des études sont menées pour voir la viabilité de solutions basées principalement sur des logiciels open source. Le projet le plus avancé est le projet « Pingo », développé par le Service public fédéral Personnel et Organisation, qui développe une station de travail basée principalement sur des logiciels open source. On peut trouver quelques informations sur ce projet sur le site Internet suivant : http://www.ulys.net/conferences/olivier.schneider.pps.

A tous les autres niveaux, des propositions de lois, de résolutions, des notes, etc. ont été déposées. En voici quelques unes :

- 3 juillet 2003, au sénat : Proposition de loi relative à l'utilisation de logiciels libres dans les administrations fédérales (déposée par Jean-François Istasse).
- 30 septembre 2003 au sénat : Proposition de loi concernant l'usage de standards ouverts et la disponibilité du code source des logiciels dans les administrations fédérales (déposée par François Roelants du Vivier et Christine Defraigne).
- 3 mars 2004, au sénat : Proposition de résolution sur l'achat de logiciels par les administrations fédérales (déposée par Jean-Marie Dedecker et Stafaan Noreilde).
- Le MET, ministère de l'équipement et des transports de la région wallonne, a décidé que tous les serveurs Internet et Intranet, si possible, seront basés sur des solutions open source. Cette information est tirée d'un interview avec Monsieur Louis Colson.
- ...

Il est d'ailleurs recommandé à tous les responsables informatiques, lors du choix d'un nouveau logiciel, de vérifier sa compatibilité avec les standards ouverts. Pour connaître les formats conseillés et proposés pour les administrations belges, il est utile de se référencer au site Internet suivant : http://www.belgif.be.

Text 4: Actions belges en faveur de l'open source

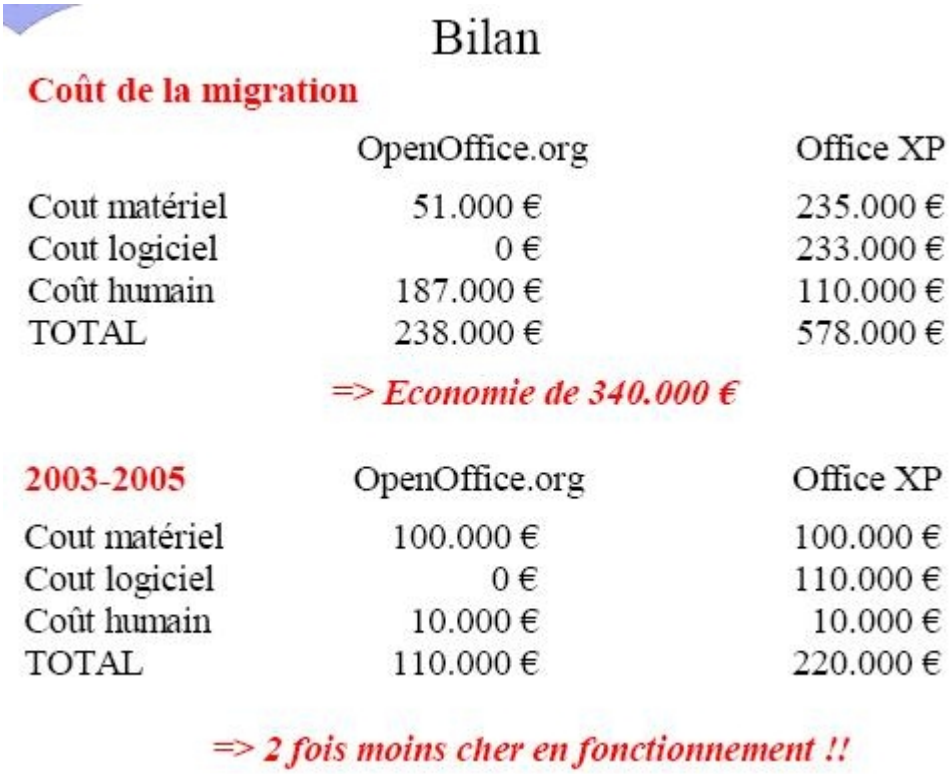
Le cas du centre hospitalier de Avranches-Granville

La Figure 10 nous montre le bilan réalisé au centre hospitalier de Avranches-Granville lors de l'étude préparatoire à la migration, de 400 ordinateurs, de la suite bureautique Microsoft Office97 vers la suite bureautique Microsoft Office Xp ou OpenOffice. Dans ce bilan, les coûts de migration et d'utilisation quotidienne sont favorable à OpenOffice.

Le centre hospitalier a aussi pris en compte, lors du choix d'OpenOffice, la facilité apportée par le au format de fichier (XML) pour l'automatisation des échanges :

- Extranet,
- Indexation automatique des documents,
- Front Office,
- Back Office.

Text 5: Le cas du centre hospitalier de Avranches-Granville extrait du site Internet : http://www.linuxplusvalue.be/mylpv.php?id=241



Bilan

Coût de la migration

	OpenOffice.org	Office XP
Cout matériel	51.000 €	235.000 €
Cout logiciel	0 €	233.000 €
Coût humain	187.000 €	110.000 €
TOTAL	238.000 €	578.000 €

=> Economie de 340.000 €

2003-2005	OpenOffice.org	Office XP
Cout matériel	100.000 €	100.000 €
Cout logiciel	0 €	110.000 €
Coût humain	10.000 €	10.000 €
TOTAL	110.000 €	220.000 €

=> 2 fois moins cher en fonctionnement !!

Figure 10: Bilan du centre hospitalier de Avranches-Granville

La gestion des licences est beaucoup plus importante pour les organisations qui produisent ou modifient des logiciels open source que pour celles qui ne font que les utiliser. En effet, l'utilisation d'un logiciel open source ne demande pas beaucoup de travail de gestion des

licences. Par contre le développement ou la modification d'un logiciel open source demande une bonne connaissance des licences utilisées. Et les obligations misent en place par certaines licences peuvent parfois rendre leur emploi inadmissible pour une organisation, par exemple si elle oblige de rendre public certaines modifications.

Il est donc nécessaire de bien connaître les licences des logiciels qu'on envisage d'employer et plus spécialement modifier. Cela permet d'éviter des désagréments inattendus.

L'enjeu économique lié au coût des licences se place donc sur le moyen et long terme. En général, ce qu'on économise avec le coût des licences est souvent dépensé dans le court terme à des formations, des tests d'intégration, à la compréhension des nouvelles licences, etc.

A long terme, l'économie réalisée sur les licences dans une grande organisation peut avoir des retombées économiques importantes. Elles permettent alors de développer des compétences internes complémentaires.

Les exemples de l'hôpital de Avranches-Granville, Text 5, et de la gendarmerie française, Text 6, sont révélateurs de l'économie qui peut être faite en utilisant des logiciels open source matures dans un environnement adéquat.

[Aboubekr & Rivard, 2005] présente le cas de la gendarmerie française qui avait lors de cette étude déjà migré 20.000 postes vers OpenOffice. Par le suite, elle en a migré 50.000 de plus. Cette migration leur a permis de réaliser des économies substantielles, environ 20 millions d'euros d'après les informations disponibles dans l'article « *La gendarmerie française fait le choix d'une bureautique libre* », [Milan].

Finalement en 2006, ils ont installés Firefox et Thunderbird comme outils de communication. De plus amples informations ces dernières migrations peuvent être trouvées dans [Bodor et Brosseau, 2006].

Text 6: Le cas de la gendarmerie française

1.3 Les enjeux technologiques liés à l'évolution des logiciels et l'innovation

Pour rappeler ce qui a été dit dans le deuxième chapitre, le développement d'un logiciel open source se fait principalement par des personnes ou des organisations qui ont besoin de cette application. Ils ont généralement le temps et l'opportunité de réaliser des programmes respectant les normes standards et les meilleures technologies mises leur disposition. Il arrive parfois qu'un logiciel open source soit complètement ou partiellement réécrit pour prendre en compte de nouvelles technologies qui rendent le logiciel plus puissant.[35]

Par ailleurs, on constate souvent qu'un logiciel open source est écrit de manière à rendre son code source lisible et bien documenté[36]. En effet, un programmeur de logiciel open source sait que d'autres personnes vont lire son code et il va mettre un point d'honneur à rendre le tout compréhensible par un autre programmeur qu'il soit expert ou débutant. C'est particulièrement vrai pour les logiciels qui atteignent un certain niveau de maturité.

35 Le logiciel Sunbird de Mozilla a subi une réécriture complète pour sa version 0.3. Le site Internet suivant offre des informations sur ce logiciel : http://www.mozilla.org/projects/calendar/sunbird.html.

36 Le logiciel Phprojekt est un exemple de logiciel documenté de manière correcte.

En outre, un projet open source dont le développement est abandonné peut facilement être repris par une nouvelle équipe qui se sent capable de continuer à le développer. Donc si nous utilisons un logiciel open source dont le développement s'arrête, nous pouvons reprendre son avenir en main et continuer à le développer suivant nos besoins propres.

Nous aussi pouvons ajouter que les logiciels open source sont très innovant car :

- ils sont développés par une communauté importante de développeurs venant de multiples horizons et que ces derniers partagent avec la communautés les dernières innovations qu'ils maîtrisent.
- leur cycle de vie est plus rapide et beaucoup plus flexible que celui des logiciels propriétaires. Et ce grâce à la vivacité de la communauté et aux faibles contraintes qu'ils subissent du monde économique.

1.4 Les enjeux humains liés à l'augmentation des compétences du personnel IT

En fonction des compétences d'une équipe IT, l'utilisation des logiciels open source peut être plus ou moins poussée. En effet, plus l'équipe IT est compétente plus elle sera capable de déployer des logiciels moins matures.

C'est pourquoi, [Guliani et Woods, 2005] classe les équipes IT en quatre catégories : les débutants, les intermédiaires, les avancés et les experts. A chaque catégorie correspond des compétences propres qui leur permettent d'utiliser et d'adopter des logiciels open source plus ou moins matures.

D'après la description de ces différentes catégories, Text 8, on constate que plus l'équipe IT est compétente, plus la société pourra utiliser des logiciels open source et les modifier en fonction de ses besoins.

Plus il y a de compétences, plus il sera intéressant de les préserver. En effet, la perte d'un expert peut poser d'énormes problèmes. Il faut donc former le personnel IT en continu, et il faut obliger les plus compétents à expliquer aux autres ce qu'ils font, à les impliquer dans leur travail et, surtout, à documenter[37] tout ce qu'ils font pour que ce ne soit pas perdu.

Signification du terme « documenter » basée sur une expérience personnelle

Lors de mon stage à EASI-WAL, il me fut demandé de modifier l'installateur d'OpenOffice 2.0 pour qu'il prenne en compte, lors de l'installation, les dictionnaires Français, Néerlandais et Allemand, à la place d'autres dictionnaires (Italien, Swahili, Thai). Pour ce faire, je dois avouer avoir au début beaucoup cherché pour comprendre comment réaliser cette demande, mais une fois le code xml de l'installateur compris, j'ai pu réaliser le travail demandé. Une fois cela terminé, j'ai rédigé un rapport sur la méthode à suivre et les endroits où modifier le code pour arriver au résultat demandé. Ainsi, lors de la sortie d'une nouvelle version d'OpenOffice, ils n'auront qu'à reprendre mon rapport et suivre la méthode décrite pour réaliser les mêmes modifications si elles sont encore nécessaires.

Text 7: Signification du terme « documenter » basée sur une expérience personnelle

37 Voir le Text 7 pour plus d'explications sur la signification de ce terme dans ce cas précis.

Nous voyons donc que plus une équipe IT est performante, plus nous pouvons employer des logiciels open source ce qui signifie une réduction des coûts liés à l'achat de licences. De plus, ils peuvent aussi profiter de leurs compétences internes pour diminuer l'utilisation de compétences externes qui coûtent souvent beaucoup d'argent à une organisation.

Les différentes catégories d'équipes IT selon [Guliani et Woods, 2005]

« Les **Débutants** ont un niveau de compétences minime. Ils ont les capacités pour faire tourner des logiciels matures tels OpenOffice, Firefox, Linux, etc. En effet, le logiciel est pour eux comme une « black box » (boîte noire) avec quelques options et fonctions simples. Les logiciels qui requièrent plus de configurations et de programmations ne leur sont pas conseillés.

Les **Intermédiaires** ont un niveau de compétences légèrement plus élevé que les débutants. Ils voient toujours les logiciels open source comme une boîte noire mais peuvent réaliser des configurations plus évoluées et utiliser de manière basique les langages de programmation.

C'est avec cette catégorie que l'on rencontre le plus de problèmes s'ils ne sont pas conscients de leur limites. Ils peuvent, par exemple, placer un logiciel open source pour une mission critique sans avoir réalisé toutes les analyses nécessaires.

Avec les débutants, les Intermédiaires représentent la majorité des équipes IT.

Les **Avancés** ne voient plus les logiciels open source comme des boîtes noires. Ils utilisent la programmation couramment pour améliorer et modifier les logiciels pour les besoins de leur société. Ils sont très puissants dans leur organisation, mais ils sont aussi à l'origine de sérieux « key-person problem ». En effet, si l'utilisation des logiciels open source à un haut niveau n'est pas correctement documentée et si les personnes moins compétentes ne sont pas formées à maintenir ces logiciels, la perte de la personne de niveau avancé peut créer de graves problèmes de fonctionnement.

Les **Experts** sont comme des stars. Ils ont la capacité de créer de nouveaux projets, de modifier la direction que prend un projet existant. Pour reprendre un phrase qui m'a marqué :

« *If it can be done with open source, expert can do it* ».

Ce n'est pas pour autant qu'un expert l'est dans tous les domaines, il ne peut l'être que pour un certain nombre de projets. Personne n'est omniscient.

Il ne faut pas s'inquiéter pour autant, un département IT peut connaître beaucoup de succès avec les logiciels open source sans pour autant arriver à un niveau expert. Il est même conseillé d'atteindre un niveau Intermédiaire et de louer les compétences nécessaires quand on en a besoin. »

Text 8: Les différentes catégories d'équipes IT

Nous pouvons aussi constater que pour utiliser des logiciels open source dans une organisation, les compétences internes requises sont plus importantes. L'utilisation de tels logiciels demande aussi une grande débrouillardise et impose aux employés de s'auto-former aux nouvelles technologies employées.

1.5 Les enjeux stratégiques liés à la non dépendance vis-à-vis d'un éditeur unique

Jusqu'à présent, l'utilisation de logiciels propriétaires, notamment les produits de Microsoft, tant pour les ordinateurs de bureau que les serveurs était presque incontournable. Maintenant, avec l'arrivée à maturité des logiciels open source, d'autres possibilités se posent à nous. En effet, selon Netcraft[38], l'utilisation d'Apache (serveur Internet) dépasse les 65% du marché pour ce type de logiciel. Et les sociétés qui proposent des solutions serveurs basées sur des logiciels open source se font de plus en plus nombreuses. On a enfin la possibilité de faire jouer la concurrence pour obtenir des services à prix concurrentiel et si le service ne correspond pas à nos attentes, il nous est toujours possible de changer de fournisseur.

Pour les ordinateurs de bureau, le marché, avec l'arrivée de logiciels comme Linux, OpenOffice, Mozilla, Thunderbird et d'autres, commence à proposer des solutions open source. Et on a donc maintenant aussi, à ce niveau, la possibilité de faire jouer la concurrence. Ce qui va permettre la diminution des prix et l'apparition de nouveaux services plus conformes aux attentes des clients.

En effet, les sociétés qui proposent des solutions open source ne peuvent faire payer que leurs compétences et les services fournis, car ils ne sont pas propriétaires des logiciels proposés. Il y a enfin moyen de faire jouer la marché soit pour avoir des meilleurs services à meilleur prix, soit pour faire descendre les prix demandés par Microsoft. Ce qui freine encore actuellement les entreprises à faire appel à ces sociétés est la garantie qu'elles offrent. Garantie quand à leur existence future et à la qualité des logiciels proposés. Mais ces garanties seront de plus en plus fortes au fil du temps et de l'utilisation des logiciels open source par un plus grand nombre.

Dans les paragraphes précédents, nous venons de voir les enjeux économiques liés à la non dépendance vis-à-vis d'un éditeur unique. Mais il existe aussi des enjeux plus politiques liés à cette non-dépendance. Nous avons maintenant aussi le choix, par exemple, des formats de fichier que nous désirons utiliser. Et notamment les formats standards ouverts qui nous évitent d'être dépendants de la politique des grands éditeurs. En effet, si Microsoft décide de modifier ses formats de fichiers et de rendre ses anciennes versions illisibles, il rendrait une partie importante de notre patrimoine numérique illisible et nous causerait d'importantes pertes. Ce qui est impossible avec les formats ouvert.

1.6 Les enjeux économiques liés aux utilisateurs

L'utilisation des logiciels open source, notamment les logiciels collaboratifs, est intéressante car ils permettent facilement d'étendre le nombre d'utilisateurs sans qu'ils ne doivent payer de licences supplémentaires, voir exemple Text 9.

En effet, avec les logiciels propriétaires, l'ajout d'un utilisateur (d'un login) oblige le contractant à payer une licence complémentaire. Ce fait limite donc l'utilisation de ces logiciels aux personnes nécessaires. Dans ce cas, peu de personnes ont accès aux données qui leur sont utiles depuis leur domicile ou un lieu de réunion sans dépenser des moyens complémentaires.

38 http://news.netcraft.com/archives/web_server_survey.html

Exemple d'utilisation d'un logiciel collaboratif open source

Comme exemple, prenons le cas d'EASI-WAL qui utilise le logiciels Phprojekt. Ce logiciel est un logiciel de gestion de projets, d'agendas, etc. en ligne. Il est basé sur la technologie php et est sous une licence open source ce qui permet de l'adapter aux besoins internes. Le choix lors de la recherche d'un nouvel outil collaboratif s'est porté sur Phprojekt car il est accessible par tous et partout. En effet, une connexion internet et un navigateur permettent de se connecter au logiciel. Ainsi, tous les collaborateurs d'EASI-WAL ont un accès aux projets, aux agendas, aux documents qui les intéressent depuis le lieu qu'ils désirent. Outre l'avantage d'y avoir accès partout, il a l'avantage d'être open source ce qui, dans ce cas, signifie gratuité d'utilisation. Ainsi chaque collaborateur peut l'utiliser sans débourser quoique se soit. Finalement, le fait de pouvoir le modifier fut aussi apprécié pour différentes raisons (autres vues, nouveaux champs, etc.).

Text 9: Exemple d'utilisation d'un logiciel collaboratif open source

2 Analyse des risques du passage à l'open source par les administrations publiques

Il existe différents risques auxquels les administrations publiques désirant passer totalement ou en partie à l'open source doivent faire attention. Les principaux risques sont :

- les risques de pérennité du logiciel liés au mode de développement de l'open source,
- les risques de sécurité liés au mode de développement de l'open source,
- les risques d'interopérabilité « organisationnelle » liés aux faibles externalités de réseau actuel de l'open source dans le monde des administrations,
- les risques liés à la maturité des logiciels open source et à la compétence de l'équipe IT,
- les risques sociaux liés au changement d'environnement de travail,
- les risques matériels liés au passage à l'open source,
- les risques logiciels liés à l'interopérabilité technique.

2.1 Les risques de pérennité du logiciel liés au mode de développement de l'open source

Comme nous l'avons vu dans la partie introduisant l'open source, son mode de développement est basé sur ce qui est appelé le mode « Bazar », à l'opposé du mode de développement des logiciels propriétaires, nommé le mode « Cathédrale ». Cette pratique de développement amène son lot de risques et de doutes.

Avec un logiciel open source, on est en droit de se demander si le logiciel qu'on utilise sera encore développé et maintenu dans 1, 2, 3 voir 4 ans. Ce questionnement est normal dans le cadre d'une vision informatique à moyen et long terme mais personne ne peut y répondre. La

meilleure façon de se faire une idée sur la viabilité du logiciel est d'analyser sa maturité[39]. Plus le logiciel sera mature, plus il sera utilisé, plus sa communauté sera importante et donc plus sa viabilité sera longue. De plus, le fait que la communauté soit soutenue par une fondation ou tout autre groupe économique est aussi un gage de pérennité de la communauté et donc du logiciel. Si malgré ces éléments le développement du logiciel et/ou de sa maintenance est abandonné, il vous est toujours possible de développer une nouvelle communauté, avec d'autres utilisateurs demandeurs, autour de ce logiciel.

Par contre, si nous décidons d'utiliser un logiciel peu mature, nous allons probablement rencontrer des problèmes de vitesse de développement et la meilleure solution dans ce cas est d'avoir les compétences qui nous permettront de participer à la communauté ou de développer une version personnelle du logiciel.

Il y a un deuxième risque lié à ce type de développement. C'est le risque lié à la vitesse et à la direction de développement du logiciel. Il est possible que la communauté n'ait pas la même vision que nous pour les développements futurs du logiciel. Dans ce cas, il y a plusieurs réponses possibles :

- soit développer les évolutions nécessaires en interne,
- soit participer à la vie de la communauté et essayer de lui faire développer les fonctionnalités nécessaires à nos besoins,
- soit créer un projet parallèle où nous définirons les développements futurs de l'application.

Ce point est important à prendre en compte, principalement, avec les logiciels peu matures. En effet, plus le logiciel est mature, plus il est facile de lui ajouter une fonctionnalité non prévue d'origine. Par exemple, un logiciel comme Firefox, qui est considéré comme un logiciel mature, permet d'ajouter facilement des fonctionnalités via des « extensions ».

Ce risque n'est pas inhérent aux logiciels libres. En effet, les éditeurs de logiciels propriétaires après un certain temps arrêtent de fournir des services pour les versions qu'ils estiment dépassées. Par exemple, Microsoft ne fournit plus de version de Office 97 qui est encore fortement employée dans les administrations belges. Ce fait oblige ces administrations à passer à une nouvelle version d'Office alors qu'elles n'en ont pas vraiment besoin. D'autant plus qu'il n'est pas toujours évident de lire un fichier Office XP avec Office 97.

2.2 Les risques de sécurité liés au mode de développement de l'open source

La sécurité en informatique fait depuis longtemps couler beaucoup d'encre[40]. Donc quand on envisage d'utiliser une solution basée sur des logiciels open source, on ne peut que se poser la question : « Les logiciels open source sont-ils aussi sûrs que les logiciels propriétaires ? ». La réponse à cette question est fortement dépendante du logiciel open source utilisé. On peut, sans se tromper, affirmer que le mode de développement de l'open source favorise la sécurité. En effet, comme les communautés développent des logiciels qui correspondent à leurs besoins et que la sécurité est un besoin important à l'heure actuelle, la sécurité est souvent mise en avant. Mais comme pour tout logiciel, il a y la possibilité de trouver des failles de sécurité

39 Le chapitre deux de [Guliani et Woods, 2005] nous informe d'avantage sur l'analyse de la maturité des logiciels.

40 Voir [AWT, 2005].

dans les logiciels open source. L'avantage qu'ont les logiciels open source face à leur concurrents propriétaires est que, si la communauté est suffisamment dynamique, les problèmes de sécurité sont vite résolus.

Il faut toutefois faire attention aux logiciels peu matures qui n'ont pas une grande communauté. En effet, la sécurité bien qu'étant un point fondamental du développement d'un logiciel open source n'est pas le plus important au début du développement. Où est l'intérêt de développer une sécurité si elle n'a rien à protéger...

2.3 Les risques d'interopérabilité « organisationnelle » liés aux faibles externalités de réseau actuel de l'open source dans le monde des administrations

Le risque d'interopérabilité au sein des administrations est élevé et dû à la faible pénétration des logiciels open source actuellement. L'utilisation de certains formats standards ouverts pose actuellement des problèmes liés à leur faible diffusion. Prenons l'exemple repris dans le Text 10 pour mieux les visualiser.

Monsieur X rédige un document avec OpenOffice, donc au format OpenDocument. Il désire l'envoyer à Madame Y qui travaille uniquement sous Microsoft Office. Lorsque Madame Y reçoit le mail de Monsieur X avec le document, elle est alors incapable de l'ouvrir. Elle est donc obligée de demander à Monsieur X de lui renvoyer le document dans un autre format. Si le document est destiné à être lu, Monsieur X peut contourner le problème en l'envoyant sous le format « pdf » qui est largement utilisé. Par contre, si le document doit être modifié par Madame Y, il sera obligé de l'enregistrer sous un format lisible par Microsoft Office, ce qui n'est pas souhaité au départ.

Dans l'autre sens, si Madame Y doit envoyer un document à Monsieur X, le problème est moindre car OpenOffice sait lire et écrire dans le format de Microsoft Office.

Text 10: Exemple simple les difficultés liées à l'utilisation du format OpenDocument

Une solution serait d'obliger Monsieur X et Madame Y à utiliser un même format de documents qui ne soit pas propriétaire, c'est-à-dire ne plus utiliser les formats de Microsoft (doc, xml, ppt, etc.). Mais Madame Y n'est pas un cas isolé à l'opposé de Monsieur X. Les formats propriétaires ont encore un longue vie dans les administrations tant qu'une politique commune sur les formats à utiliser n'est pas dégagée et appliquée. Politique qui est actuellement débattue dans les différents niveaux de pouvoirs en Belgique[41].

Une autre solution serait de faire passer toutes les administrations à OpenOffice...

2.4 Les risques liés à la maturité des logiciels open source et à la compétence de l'équipe IT

Il est très important d'analyser la maturité d'un logiciel open source avant d'envisager de l'utiliser. Les logiciels comme OpenOffice, Firefox, Thunderbird et Linux sont considérés comme matures. Ce qui signifie que les risques liés à leur déploiement sont moindre que pour

41 Voir : Text 4, page 46 concernant les actions belges en faveur de l'open source.

les autres logiciels. Mais les utiliser comporte aussi certains risques au moment d'une migration (perte de données, incompatibilité des macros Office et OpenOffice, etc.). Mais mettre en place une solution composée de logiciels matures ne demande pas un personnel IT avec des compétences avancées.

Lorsqu'on envisage d'utiliser des logiciels open source moins matures, les risques augmentent. En effet, on risque de rencontrer les mêmes difficultés qu'avec un logiciel mature mais il faut dans ce cas ajouter le risque lié à la compétence du personnel IT. Un personnel IT ayant un niveau de compétence avancé ou expert ne rencontrera probablement pas de problèmes excessifs à mettre en place une solution basée sur des logiciel peu matures. Mais un personnel ayant des compétences intermédiaires pourrait se croire compétent pour le travail alors qu'il ne l'est pas toujours. Il risque alors de mettre en production des logiciels qu'ils pensent être corrects et efficaces alors qu'ils ne le sont pas encore.

Il est donc nécessaire de connaître le niveau de maturité des logiciels envisagés et de connaître le niveau de compétences du personnel IT. Si nous voulons éviter certains risques, n'hésitons pas à faire appel à des sociétés externes qui pourront analyser les différents éléments pour nous. Ainsi, notre personnel IT pourra se former à leur contact et acquérir des compétences nouvelles exploitables par la suite.

2.5 Les risques sociaux liés au changement d'environnement de travail

Le principal risque social est le refus du nouveau système. Ce refus peut être plus ou moins fort, refus total ou partiel. Dans les deux cas, on peut dire que la migration sera un échec. Pour éviter cet échec, il faut bien préparer les utilisateurs à la migration.

Tout utilisateur a peur de ce qu'il ne connaît pas, du changement. On observe cette réaction dans d'autres domaines que celui de l'informatique. On peut, d'ailleurs, comparer une migration à l'achat d'une nouvelle voiture. Au début, dans les deux situations, on est un peu perdu mais si on nous explique clairement et calmement comment tout fonctionne, on finira par dominer son nouvel environnement.

Il ne faut jamais oublier que l'utilisateur est l'acteur principal du changement. Il ne faut donc pas négliger le facteur humain. Voici quelques pistes à suivre pour éviter de tels problème lors d'une migration :

- expliquer aux utilisateurs finaux les avantages que la migration va leur apporter,
- développer une vision,
- mettre en place des personnes ressources,
- proposer des formations.

2.6 Les risques matériels liés au passage à l'open source

Le risque d'incompatibilité matériel peut avoir des conséquences désastreuses sur le plan économique s'il n'a pas été étudié dès le début des tests de migration. En effet, si tous les ordinateurs possèdent un composant (ex : une carte graphique, une carte mère, etc.) non reconnu par le nouveau système, il est intéressant de le savoir avant de commencer la migration.

Il existe deux méthodes principales pour contourner ce risque :

- Soit on change le matériel avec du matériel compatible. Ce qui peut coûter assez cher dans certaines situations.
- Soit les informaticiens internes ou un sous-traitant développent des drivers pour ce matériel. Dans certains cas, c'est probablement la solution la moins onéreuse.

Si sur un parc informatique de 10000 machines, il y en a 20 qui ont des problèmes de compatibilité avec le nouveau système, il est certainement plus intéressant d'acheter du nouveau matériel. Mais si 1000 machines ont le même problème d'incompatibilité, il est peut-être intéressant de développer le driver utile à leur bon fonctionnement. Chaque cas a sa propre solution. Et dans certaines situations, la meilleure solution est parfois de ne pas migrer vers GNU/Linux.

Il est à noter que les sociétés productrices de matériel informatique proposent de plus en plus souvent une version de leurs drivers pour Linux. C'est, en effet, pour elles un marché qu'il convient de gagner.

Il faut donc, avant une migration, connaître le parc informatique qui accueillera le nouveau système et vérifier que le matériel sera compatible. Sinon, on risque de rencontrer des surprises qui peuvent coûter cher.

2.7 Les risques logiciels liés à l'interopérabilité technique

Le risque logiciel est probablement le plus important et le plus difficile à résoudre. En effet, toute société a des logiciels fait « maison » qui correspondent à des besoins internes. Ces logiciels n'ont pas toujours été programmés en utilisant les standards, lors de migration, cela peut créer d'énormes problèmes. Dans beaucoup de cas, ces logiciels ne fonctionnent pas dans le nouveau système. Il est alors nécessaire de trouver une nouvelle solution qui consiste :

- soit à trouver sur le marché un autre logiciel qui pourrait convenir,
- soit réécrire le logiciel pour qu'il fonctionne sur le nouveau système.

Ces deux solutions peuvent parfois coûter très cher.

Il y a aussi un risque important lié aux logiciels communicants. Par exemple, un logiciel développé pour communiquer avec Microsoft Office ne fonctionnera probablement pas avec Open Office. Dans ce genre de situations, une solution devra être trouvée avant tout changement.

Quand un logiciel pose problèmes, il faut se poser quelques questions :

- Est-ce que ce logiciel était primordial pour le fonctionnement de l'entreprise ?
- Quel âge a-t-il ? Ne peut-il pas être dépoussiéré et modifié pour respecter les standards ?
- Peut-il être facilement remplacé par un autre qui a les mêmes fonctionnalités ?

Si après avoir répondu à ces questions pour chaque logiciel posant problème, on constate qu'il est possible de trouver, dans la majorité des cas, une solution économiquement et techniquement viable alors on peut envisager de poursuivre la migration.

Si par contre, on constate qu'il est impossible de remplacer à un coût correct un certain nombre de logiciels, il faut alors envisager de mettre nos projets de migration en suspens, le temps de trouver un autre plan de migration qui pourrait tenir compte de ces problèmes.

Tout problème peut, avec de la motivation, trouver une solution. Développer en interne des logiciels de remplacement, en développer un avec l'aide de la communauté Open Source, etc.

Les deux risques ici sont donc :

- Le risque le plus grave est de ne pas s'être rendu compte d'incompatibilités logicielles avant le déploiement du nouveau système. Dans ce cas, il faut parfois faire marche arrière et les frais sont souvent importants.
- Le risque le moins grave est de constater des incompatibilités logicielles lors de la phase de test. Cela va probablement entraîner des retards dans la migration et donc une perte d'argent mais moindres que si l'on se retrouvait dans la situation ci-dessus. Dans ce cas, on peut réfléchir tranquillement à des solutions.

Dans beaucoup de migrations, on rencontre le problème d'incompatibilités logiciels. Il est donc fortement recommandé de tester tous les logiciels avant le déploiement massif du nouveau système. On évitera ainsi de se retrouver avec un système non-fonctionnel, et des frais importants causés par ce problème.

Chapitre 4 : Évaluation de l'existant en logiciel open source

Ce chapitre va consister à comparer certains logiciels propriétaires et les logiciels open source qui pourraient avantageusement les remplacer. Nous avons du limiter notre démarche à certaines catégories de logiciels. Ces catégories sont : les systèmes d'exploitation, les suites bureautiques, les navigateurs Internet, les clients email et les groupwares ou logiciels collaboratifs.

Les logiciels propriétaires de référence seront des logiciels utilisé à la Région Wallonne.

Ce chapitre sera divisé en trois principaux points, le premier va proposer une définition des principales catégories de logiciels analysées.

Le deuxième point proposera un tableau recensant des logiciels propriétaires et leurs correspondants open source.

Le dernier point va par catégorie de logiciel comparer la solution propriétaire à des solutions open source.

1 Les différentes catégories de logiciels analysées

1.1 Système d'exploitation ou « Operating system »

C'est un ensemble de logiciels qui mettent en relation les logiciels de haut niveau, logiciels utilisateurs, et la couche physique de l'ordinateur.

Il assure le démarrage et la fermeture de l'ordinateur. De plus, il fournit des interfaces standardisées aux logiciels de haut niveau pour interagir avec les différents périphériques reliés à l'ordinateur.

Les systèmes d'exploitations les plus connus sont ceux de Microsoft, d'Apple et les systèmes Unix et dérivés.

Les différentes catégories de logiciels analysées

1.2 Suite bureautique

C'est un ensemble de logiciels comprenant souvent un traitement de texte, un tableur, un logiciel de présentation, un base de données, un logiciel de mise en page de sites Internet, etc. En fait, tous logiciels utiles à un travail de bureau.

Les logiciels les plus connus sont les suites Office de Microsoft, StarOffice de Sun Microsystems et OpenOffice.

1.3 Navigateur Internet

C'est un logiciel conçu pour interpréter les pages du « World Wide Web ». Il est composé, principalement, d'un moteur de rendu des standards Internet et d'une interface utilisateur.

Les navigateurs les plus connus sont Netscape, Internet Explorer, Safari, Opera et Mozilla Firefox.

1.4 Client email ou client de courrier électronique

C'est un logiciel comparable à une boîte aux lettres. Il permet, principalement, d'envoyer et de recevoir des mails.

Les clients de courrier électronique les plus connus sont Eudora, Microsoft Outlook et Express, Mail et Mozilla Thunderbird.

1.5 Groupware ou logiciel collaboratif

« *Le groupware est l'ensemble des technologies et des méthodes de travail associées qui, par l'intermédiaire de la communication électronique, permettent le partage de l'information sur un support numérique à un groupe engagé dans un travail collaboratif et/ou coopératif* »
Jean-Claude Courbon[42].

Certains des logiciels les plus connus sont ContactOffice, Lotus Notes, Novell Group Wise, Microsoft Project, OpenGroupware, eGroupware et Phprojekt.

2 Recension des logiciels propriétaires utilisés à la RW et leur équivalent open source

	Propriétaire	**Open Source**
Système d'exploitation	Microsoft Windows	GNU/Linux
Suite bureautique : • **traitement de texte** • **tableur** • **logiciel de présentation** • **base de données** • **autres**	Microsoft Office : • Word • Excel • Power Point • Acces • Outlook	OpenOffice : • Writer • Calc • Impress • Base • Draw

42 Université de Genève

	Propriétaire	Open Source
Navigateur Internet	Internet Explorer	Firefox
Client email	Microsoft Office Outlook	Thunderbird
Groupware	MS Project Teamware	• Evolution • OpenGroupware • Phprojekt • eGroupware • Grouwise • ... (évolution constante et pas de logiciel qui a fait ses preuves comme OOo)

3 Comparaison des solutions au point de vue technique et financier

Dans les différents points suivants, nous allons essayer de proposer des points de comparaison entre les différents logiciels du tableau ci-dessus. Mais il est évident que seul un essai personnel des différentes solutions permet de voir lesquels nous conviennent le mieux.

3.1 Système d'exploitation

Il est difficile de comparer au niveau technique les différents systèmes d'exploitations existants. Ils ont tous certaines particularités et fonctionnalités qui les rendent intéressants. Par exemple, Windows est considéré comme le système d'exploitation le plus convivial et les systèmes basés sur GNU/Linux et les différents UNIX sont considérés comme les plus paramétrables.

La meilleure solution est donc de lire de la documentation spécialisée sur chacun des systèmes d'exploitations disponibles sur le marché et bien entendu de les essayer pour trouver celui qui conviendra le mieux aux besoins recherchés.

Au niveau financier, il existe des solutions basées sur GNU/Linux gratuites ce qui n'est pas le cas de Windows. Mais il existe aussi des solutions basées sur GNU/Linux payantes qui fournissent généralement un service de soutien complémentaire.

Pour nous y retrouver parmi les multiples propositions, *Linux ou Windows un guide d'aide à la décision* de Philippe Logerot, [Logerot, 2003], présente les solutions basées sur GNU/Linux les plus en utilisés et les différentes solutions proposées actuellement par Microsoft. Il reprend tant les avantages techniques des différentes solutions qu'une comparaison de leur prix.

Nous conseillons en plus de visiter les sites Internet des différents leaders du marché :

- Les produits Microsoft Windows : http://www.microsoft.com/france/windows/default.mspx,
- Les produits Red Hat : http://www.redhat.com/,
- Les produits SuSE : http://www.novell.com/linux/,
- Les produits Debian : http://www.fr.debian.org/,
- Les produits Ubuntu : http://www.ubuntu-fr.org/.

Text 11: Système d'exploitation : liens utiles

3.2 Suite Bureautique

Au niveau des prix, il est évident que OpenOffice est plus intéressant grâce à sa gratuité et aux prix élevés des différentes versions de Microsoft Office. Pour avoir une idée des prix proposés par Microsoft, il d'aller regarder la liste officielle des prix qu'il propose chaque mois[43].

Au niveau technique, Microsoft a encore quelques longueurs d'avance par rapport à OpenOffice; cette longueur ayant fortement diminué depuis la sortie d'OpenOffice 2.0. Le fait que nous voyons OpenOffice comme un clone de Microsoft Office nous fait malheureusement souvent oublier les nouveautés qu'il apporte et sa relative jeunesse.

A la différence de Microsoft Office qui n'offre que des versions pour Windows et pour Mac, OpenOffice est disponible sur un grande majorité de plateformes existantes (Windows, Unix, Mac, Linux). Et son format standard, OpenDocument, est un format ouvert et donc utilisables par d'autres logiciels sans perte de données ce qui n'est pas le cas des formats standards de Microsoft Office.

A l'heure actuelle, il est difficile de trouver des documents comparant, de manière non partisane, les deux solutions. Les documents comparatifs référencés ci-dessous ainsi que les sites Internet de référence[44] peuvent toutefois nous éclairer sur différents aspects des deux solutions.

43 La liste des prix officiels de Microsoft peut être récupérée via le site Internet suivant : http://members.microsoft.com/partner/france/tarif/tarif_ms/default.aspx.

44 Sites Internet de référence pou les suites bureautiques :
- Microsoft Office : http://office.microsoft.com/fr-be/default.aspx.
- OpenOffice : http://www.openoffice.org/ ou http://fr.openoffice.org/.

Articles intéressants à lire sur les suites bureautiques :

- Passer de Microsoft Office 2003 à OpenOffice 2 ? (http://www.lesnumeriques.com/article-229.html, 22/04/2006)
- Ortograf: OpenOffice vs Microsoft, Jean Véronis (http://aixtal.blogspot.com/2005/11/ortograf-openoffice-vs-microsoft.html, 22/04/2006)
- OpenOffice.org or Microsoft Office (http://www.matt13.com/computer/open_office_or_ms_office/index.html, 22/04/2006)
- MS Office, OpenOffice, StarOffice : Word, Writer, StarWriter (http://www.help-info.net/comparatif-6-microsoft-office-open-office-star-office-word-writer-starwriter.html, 22/04/2006)
- Office ou OpenOffice : La course fonctionnelle a démarré ! (1ère partie), Bertrand Garé, L'informaticien n°025 (http://www.itrmanager.com/46485-office,open,office,course,fonctionnelle,demarre,1ere,partie.html, 22/04/2006)
- OpenOffice.org stabilise la version 2.0 (2ème partie), Bertrand Garé, L'informaticien n°025 (http://www.itrmanager.com/46584-open,office,org,stabilise,version,2,0,2eme,partie.html, 22/04/2006)
- OpenOffice.org stabilise la version 2.0 (3ème partie), Bertrand Garé, L'informaticien n°025 (http://www.itrmanager.com/46633-open,office,org,stabilise,version,2,0,3eme,partie.html, 22/04/2006)

Text 12: Suite Bureautique : liens utiles

3.3 Navigateur Internet

Les navigateurs Internet, Internet Explorer (IE) et Firefox, sont tous deux « gratuits » soit fourni avec Windows pour le premier soit en téléchargement sur le site Internet de Mozilla pour le second.

Du point de vue technique, tout le monde s'accorde à dire que Firefox est plus avancé qu'Internet Explorer 6. D'ailleurs, la future version 7 d'Internet Explorer reprendra plusieurs éléments qui font le succès de Firefox comme les onglets.

Ce qui est déroutant est la limitation envisagée par Internet Explorer 7 qui n'offrira toutes ses nouveautés et ses possibilités que sous le nouveau Windows Vista. Au contraire de Firefox qui est indépendant de la plateforme utilisée (Windows, Mac, Linux ou Unix).

Pour avoir une meilleure idée des deux solutions, nous conseillons de visiter leur site de référence[45] et les articles suivants proposant des comparaisons entre IE et Firefox.

45 Site Internet de référence pour les navigateurs Internet :
- Internet Explorer : http://www.microsoft.com/windows/ie_intl/fr/default.mspx,
- Mozilla Firefox : http://www.mozilla-europe.org/fr/products/firefox/

Articles intéressants à lire sur les navigateurs Internet :

- Comparatif de navigateurs : Internet Explorer, Firefox et Opera (http://www.teki.info/Dossier/comparatif-internet-explorer-firefox-opera.php, 22/04/2006)
- Comparatif Internet Explorer / Firefox, Xavier Mortelette (http://www.supinfo-projects.com/fr/2005/comparatif_ie_ff/, 22/04/2006)
- Internet Explorer Versus Mozilla Firefox, Emmanuel Clement (http://emmanuel.clement.free.fr/navigateurs/comparatif.htm, 22/04/2006)
- Internet Explorer contre Firefox, Robert Vamosi (http://www.zdnet.fr/produits/logiciels/internet/0,39049754,39184453,00.htm, 22/04/2006)
- Comparatif internet explorer mozilla firefox (http://www.adojeunz.com/ados/article0024.html, 22/04/2006)

Text 13: Navigateur Internet : liens utiles

3.4 Client Mail

Au niveau des prix, à nouveau, nous pouvons dire que Thunderbird est gagnant par rapport à Outlook qui nécessite de payer une licence soit séparée soit avec la suite bureautique Office. Par contre, Outlook Express est fourni directement avec Windows.

Thunderbird a les mêmes fonctionnalités de bases qu'Outlook Express c'est-à-dire la gestion du courrier électronique (envoi et réception de e-mail et un gestionnaire de contact). Il propose aussi comme Outlook un agenda grâce au plug-in « Lightning » ou via le logiciel Sunbird.

Toutefois, dans le milieu professionnel, Outlook est souvent considéré comme étant sans équivalent en open source. C'est principalement lié au fait qu'Outlook en plus de gérer la messagerie électronique, intègre aussi : un agenda et un gestionnaire de tâches.

Pour ce qui est des considérations purement techniques, nous conseillons les sites de référence[46] et les articles suivants qui comparent les deux logiciels.

46 Sites de référence des clients mail :
Outlook : http://www.microsoft.com/france/office/outlook/prodinfo/.
Outlook Express : http://www.microsoft.com/france/internet/produits/outlook/default.mspx.
Thunderbird : http://www.mozilla-europe.org/fr/products/thunderbird/.
Sunbird : http://www.mozilla.org/projects/calendar/sunbird/index.html.
Lightning : http://wiki.mozilla.org/Calendar:Lightning.

Articles intéressants à lire sur les clients mail :

- Outlook Express ou Thunderbird: quel logiciel de messagerie choisir?, Gilles Rocard (http://www.zdnet.fr/entreprise/management-rh/collaboratif/0,50007183,39332480,00.htm, 22/04/2006)
- Quel logiciel de messagerie choisir? (http://www.arobase.org/softs/choisir.htm, 22/04/2006)
- Le client de messagerie Thunderbird au crible (http://solutions.journaldunet.com/0412/041209_thunderbird.shtml, 22/04/2006)
- Outlook vs Thunderbird : 1 partout, Yannick Guerrini (http://www.presence-pc.com/actualite/faille-thunderbird-14100/, 22/04/2006)

Text 14: Client Mail : liens utiles

3.5 Groupware

En ce qui concerne les groupware, il est beaucoup plus difficile de les comparer car il en existe une plus grande variété. Et chacun a des fonctionnalités qui le rende différent des autres.

De plus, il existe des solutions payantes comme MS Project de Microsoft, des solutions open source pour lesquelles il existe des services payants et d'autres qui ne proposent que le logiciel, en open source, avec au minium des tutoriels d'installation.

Articles de comparaison entre groupware ou les sites Internet officiels de certains grouware :

- Solutions Groupware : Etat des lieux, Jérôme Calais (http://linuxfr.org/2004/01/09/15039.html, 22/04/2006)
- Les outils de groupware en ligne (http://www.outilsfroids.net/news/105.shtml, 22/04/2006)
- MS Office Project 2003, site Internet officiel (http://www.microsoft.com/france/office/project/default.mspx, 22/04/2006)
- Novell GroupWise, site Internet officiel (http://www.novell.com/fr-fr/products/groupwise/, 22/04/2006)
- phpGroupWare, site Internet officiel (http://www.phpgroupware.org/, 22/04/2006)
- OpenGroupware.org, site Internet officiel (http://www.opengroupware.org/, 22/04/2006)
- eGroupWare, site Internet officiel (http://www.egroupware.org/, 22/04/2006)
- Phprojekt, site Internet officiel (http://www.phprojekt.com/, 22/04/2006)
- ContactOffice, site Internet officiel (http://www.contactoffice.com/, 22/04/2006)

Text 15: Groupware : liens utiles

Chapitre 5 : Travail préparatoire à la réalisation d'un guide d'étapes pour une migration vers l'Open Source

Ce chapitre a pour objectif dans un premier temps de présenter ce qu'est une migration et ses différents types.

Ensuite, nous proposons quelques études de cas de migrations vers l'open source qui se sont déroulées dans différentes administrations européennes.

Finalement en nous basant sur ces études de cas et sur l'analyse de différents guides de migrations, nous proposerons les étapes qui seront à la base de notre propre guide de migration détaillé dans le chapitre suivant.

1 Qu'est ce qu'une migration ?

La migration consiste à quitter un environnement pour un autre. Ce terme est couramment utilisé pour la migration de populations. Nous pouvons donc comparer une migration informatique à une migration de population qui consiste pour une personne à quitter son pays pour se rendre dans un autre. Il change donc d'environnement. Dans le monde du travail, un déménagement peut être considéré comme une migration car il entraîne un changement d'environnement. En informatique, une migration signifie changer l'environnement informatique. Cette migration a lieu quand les décideurs estiment que le système informatique ne répond plus correctement à leur attente aux niveaux économiques, techniques et productifs.

Pour beaucoup de spécialistes, une migration est considérée comme un événement pouvant engendrer des troubles psychologiques. En effet, les utilisateurs du système se trouvent face à un nouvel environnement qu'ils ont parfois difficile à comprendre. L'informatique a sa logique qui n'est pas celle des humains. C'est pourquoi il est nécessaire d'accompagner les utilisateurs dans la migration.

Au niveau technique, une migration consiste à modifier la configuration logicielle du système. Cette modification peut être légère ou conséquente. On parle généralement de migration lorsqu'on change le système d'exploitation des ordinateurs. Par exemple, le passage de Windows 98 à Windows XP consiste en une migration tout comme le passage de Windows à GNU/Linux.

Mais une migration peut simplement consister dans le fait de passer d'une version d'un

logiciel à une autre. Par exemple, passer de Microsoft Office 97 à Microsoft Office XP. Entre ces deux types de migrations, on trouve toutes les migrations qui consistent au changement d'un logiciel par un autre. Par exemple, passer de Microsoft Office à Open Office ou de Outlook à Thunderbird.

Dans les différents exemples de migrations citées, nous pouvons constater que la préparation à la migration doit se faire de manière différente en fonction de la modification envisagée. En effet, passer de Microsoft Office 97 à Microsoft Office XP demande moins de tests de validité et de formations pour les utilisateurs que pour passer de Microsoft Office à Open Office. Dans la même optique, passer d'une version à une autre de Windows demandera moins d'efforts que de passer de Windows à GNU/Linux qui a un système de fonctionnement totalement différent. Dans ce cas, ce n'est pas uniquement les utilisateurs qui devront faire connaissance avec un environnement qui leur est totalement inconnu mais aussi les informaticiens qui devront apprendre à maîtriser leur nouvel environnement de travail. Donc ce genre de migration ne s'improvise pas. C'est pourquoi, par la suite, je proposerai des étapes à suivre pour éviter de tomber dans certains pièges qui peuvent coûter cher à l'entreprise ou à l'administration qui décide de migrer son système informatique.

La migration vers une solution basée sur les logiciels open source est un grand défi tant au niveau technologique qu'au niveau humain. Mais si elle réussit, elle peut apporter beaucoup Comme nous l'avons vu dans le premier point du troisième chapitre consacré aux enjeux lié à l'open source.

1.1 Différents types de migrations

Il est possible de les définir par trois termes. Les deux premiers termes ont été repris de [Aboubekr & Rivard, 2005] et le troisième a été défini après lecture de différents documents parlant de migration informatique.

- Le premier est dû à la politique menée par les états. En effet, les états se prononcent de plus en plus vis-à-vis des logiciels open source. On peut en dégager trois catégories. Ceux qui imposent la migration à leurs administrations, ceux qui se posent en faveur des logiciels open source et ceux qui évitent de prendre position pour diverses raisons.
- Le second vient du fait que l'administration ou la société peut décider de migrer en douceur ou rapidement. La migration en douceur désigne un migration qui se fait en plusieurs étapes. La migration rapide consiste en une migration où, du jour au lendemain, on passe de l'ancien au nouveau système.
- Le troisième vient du fait que l'on peut réaliser une migration partielle ou totale. La migration partielle consiste à changer une ou plusieurs applications mais pas toutes. La migration totale consiste à migrer toutes les applications dont le système d'exploitation.

Une migration en douceur et totale peut être réalisée par une succession de migrations partielles.

1.1.1 Migration imposée vs Migration volontaire

Ces deux types de migrations dépendent de choix politiques nationaux ou d'entreprises. En effet, dans certains pays, l'État, via des lois, des décrets ou des règlements, impose à ses administrations de passer à des solutions open source. Dans d'autres, l'État s'est déclaré

favorable à l'utilisation des logiciels open source mais n'impose actuellement rien à ses administrations. D'autres encore ne se prononcent ni pour l'un ni pour l'autre pour des raisons politiques évidentes.

La migration imposée

Pourquoi promulguer des lois, des décrets ou des règlements imposant l'utilisation des logiciels open source à ses administrations ? Nous avons pu constater que la plupart des pays où l'État « impose » l'utilisation de logiciels open source sont des pays en voie de développement (Venezuela, Pérou, Corée du Sud). Dans ces pays, épargner de l'argent sur l'achat des licences est vital pour l'image de marque du pouvoir vis-à-vis du peuple. Et, l'argent épargné est ainsi consacré à des projets de première nécessité.

Pourquoi parler de l'image de marque ? Le peuple ne comprend pas pourquoi donner de l'argent à des multinationales alors que le peuple a d'autres besoins que l'informatique. De plus, le développement de ces technologies permet de créer des emplois et un savoir faire qui, dans le futur, pourra s'exporter. Et finalement apporter un meilleur niveau de vie à la population.

Il est évident que l'utilisation de logiciels propriétaires n'est pas interdite si c'est la meilleure solution disponible sur le marché. La migration vers les logiciels open source se fait, en général, en douceur. Par exemple en Corée du Sud, la migration consiste, d'ici 2007, à remplacer progressivement les logiciels propriétaires par des logiciels open source dans les administrations dépendantes du gouvernement.

La migration volontaire

Ces pays ne promulguent pas de lois, ni de décrets, ni de règlements visant à utiliser des logiciels libres mais promeuvent des solutions open source. Pour cela, ils réalisent des recherches et des études de faisabilité, et proposent de l'aide aux personnes qui désireraient en savoir plus. On peut dire qu'ils ont une démarche pro-active en faveur des logiciels open source. Pour ne citer que deux exemples :

Le Brésil propose à ses administrations de passer sous « Freedows », un logiciel qui permet de faire tourner les applications Windows et Linux, avant de migrer ses administrations à moyen terme vers Linux.

Le Japon, la Chine et La Corée du Sud développent en commun un système d'exploitation baptisé « Asinux » qui se base sur Linux.

Ces pays sont donc de grands partisans des logiciels open source car ils proposent aux administrations d'épargner de l'argent et de le redistribuer là où il est nécessaire.

De plus, l'utilisation des logiciels open source permet l'accès à l'informatique au plus grand nombre. Par exemple, l'administration Brésilienne met en place des ordinateurs sous Linux dans ses écoles.

1.1.2 Migration en douceur vs Migration rapide

Ces deux types de migration dépendent du choix politique fait au sein de l'administration ou de la société qui décide de migrer. Ce sont deux approches totalement différentes. Pendant

que les uns préparent une migration étalée dans le temps, d'autres choisissent une migration radicale et rapide.

Migration en douceur

Cette migration, qui a mes faveurs, a pour objectif de réaliser une migration en plusieurs phases. Par exemple, beaucoup de ces migrations passent, dans un premier temps, par la migration des outils bureautiques, ce qui permet aux utilisateurs de s'adapter à leurs nouveaux outils. Dans un second temps, il est envisagé de passer à Linux. Comme les utilisateurs sont déjà adaptés à leurs nouveaux outils, ils ne voient presque pas le changement de système d'opération. Et le temps de formation est réduit car les utilisateurs ne doivent apprendre qu'une chose à la fois et leur monde n'est pas complètement bousculé. Ils ont donc le temps de s'adapter à ces changements successifs et la perte de productivité est moindre (temps d'adaptation diminué).

Migration rapide

Cette migration consiste au passage de l'ancienne solution à la nouvelle sans étapes intermédiaires. Ces migrations sont réalisées pour des raisons financières, de qualité, de sécurité, de protection contre les virus, etc. Elles nécessitent l'engouement des utilisateurs, sinon, la migration risque d'être un fiasco (mauvaise volonté des utilisateurs, perte radicale de productivité). Les utilisateurs doivent avoir la possibilité de s'adapter à leur nouvel environnement. Des formations doivent être proposées.

D'après une expérience personnelle[47], le passage de Windows à Linux pour des applications uniquement bureautiques peut se faire sans difficultés si l'utilisateur est correctement suivi et s'il est motivé par ce changement.

1.1.3 Migration partielle vs Migration totale

Migration partielle

Une migration partielle est souvent une étape vers une migration totale. Son objectif est de remplacer un ou plusieurs logiciels. Ainsi les utilisateurs ne doivent s'adapter qu'à quelques nouveautés mais pas au système entier. Une migration partielle peut consister à passer de Microsoft Office à Open Office mais aussi, de mon point de vue, de Windows à Linux si on utilisait déjà sous Windows la majorité des applications que l'on va continuer à utiliser sous Linux.

Dans une migration partielle, le temps d'adaptation et les temps de formation sont réduits. C'est une méthode pour étaler dans le temps les coûts liés à la migration. Elle permet en outre de valider certains choix techniques. Et si ces choix ne sont pas adaptés, il est toujours permis de faire marche arrière à moindre coût.

47 En 2004, l'Assemblée Générale des Etudiants de Namur a fait migrer l'ordinateur du permanent d'une solution Windows à une Solution Linux sans qu'il rencontre de difficulté d'adaptation aux nouveaux outils proposés (OpenOffice, Firefox et Thunderbird étant les principaux logiciels utilisés après la migration).

Migration totale

Une migration totale consiste au changement complet du système. Changement du système d'exploitation et des applications. Par exemple passer d'un ordinateur sous Windows avec Microsoft Office à un ordinateur sous GNU/Linux avec Open Office. Il est possible de prévoir une migration totale comme une suite de migrations partielles. Donc comme une migration totale en douceur. On voit aussi des migrations totales rapides mais elles comportent généralement plus de risques de ne pas être adoptées par les utilisateurs.

	Migration partielle	***Migration totale***
Migration en douceur	• Bonne solution quand il s'agit d'une série de logiciels beaucoup employés par les utilisateurs finaux (ex : laisser le choix pendant une certaine période entre l'ancienne et la nouvelle solution). • Mauvaise solution quand cela concerne des logiciels qui n'ont aucun impact sur l'environnement de travail des utilisateurs finaux (ex : bases de données, serveur mail, etc.).	• Très bonne solution qui permettra aux utilisateurs de se faire à leur nouvel environnement de travail.
Migration rapide	• Ni bonne, ni mauvaise solution il s'agit d'une série de logiciels beaucoup employés par les utilisateurs finaux • Mauvaise solution quand cela concerne des logiciels qui n'ont aucun impact sur l'environnement de travail des utilisateurs finaux	• Solution envisageable si les utilisateurs sont demandeurs de la migration. • Sinon mieux vaut éviter ce type de migration.

Table 2: Tableau récapitulatif des différentes possibilités de migration

2 Études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source

De par le monde, il existe toute une série d'expériences d'administrations publiques et de sociétés, dans le sens large du terme, qui ont réalisé une migration vers l'open source réussie ou pas et ce dans différentes proportions.

Nous allons citer quelques exemples de migrations, leur étendue et leur réussite. Nous allons être principalement attentifs aux points suivants :

- le contexte ayant poussé à la migration,
- la méthode utilisée,
- le niveau de réussite,
- ce qui aurait pu être fait différemment après retour d'expérience.

2.1 Expériences Internationales et Nationales

Nous allons commencer par voir deux initiatives internationales puis nous verrons trois initiatives nationales.

Les deux expériences internationales, Gdańsk et Estrémadure, et l'expérience de la ville de Schoten ont été reprises de différentes sources[48] dont l'IDABC. Nous allons les présenter de manière résumée.

L'expérience de la ville de Ciney est présentée sur base d'une conférence de Monsieur D. Delaunois, en 2005 aux Facultés Universitaires Notre Dame de la Paix, Namur, et sur base d'une interview de cette même personne.

L'expérience du commissariat EASI-WAL est basé sur des entretiens avec les membres de son personnel et plus particulièrement ses responsables et sur une enquête auprès des membres su personnel lors de mon stage sur place. Le questionnaire de l'enquête est présenté dans le troisième point de l'annexe.

2.1.1 Internationales

Gdańsk

Gdańsk est située dans la province de Porémanie (Pomorskie) et est la plus grande ville maritime de Pologne. Elle a une population estimée à environ un demi-million d'âmes.

La migration vers des logiciels open source s'est faite en plusieurs étapes et pour les différentes raisons suivantes : la sécurité, la fiabilité, la stabilité et la réduction des coûts.

Méthode utilisée

En 2001, l'administration a migré ses services mails de MS Exchange sur Windows NT à une solution basée sur le système d'exploitation Linux, RedHat, et sur le serveur mail Postfix.

En 2002, ils ont remplacé leurs serveurs Internet sous Windows NT par des serveurs Apache sous Linux, RedHat.

48 Ces sources sont référencées dans le Text 16, le Text 17 et le Text 18.

Figure 11: Pologne - Gdańsk

Ils ont aussi mis en place un FireWall basé sur Linux, RedHat.

Leur département d'architecture a migré ses bases de données Oracles vers PostgreSQL 7.2 sous Linux, Debian. Par contre, la migration des bases de données d'autres départements n'a pu se faire à cause de problèmes techniques. Ils ont donc décidé de mettre en place Oracle sous Linux ce qui d'après eux est la solution optimale pour les besoins de leur ville.

Le service incendie a remplacé la suite Office de Microsoft par OpenOffice. Au niveau technique, tout s'est fait sans soucis, une version polonaise existant et la compatibilité avec les applications de Windows étant jugée suffisante, mais il a fallut former les utilisateurs à cette nouvelle suite bureautique.

Niveau de réussite

Ils ont, grâce à l'open source, pu diminuer leurs dépenses en informatique. Ils sont maintenant indépendants de la « dictature » imposée par les vendeurs. Ils peuvent donc investir à leur rythme et dans ce qui leur est nécessaire. Ils ont aussi constaté le manque de l'open source dans certains cas (ici : les bases de données principalement).

Ils ont tiré de cette migration certains points positifs et négatifs :

- Positif :
  - accès libre et gratuit à l'utilisation des logiciels open source,
  - grande variété de logiciels open source,
  - le code source étant ouvert, il permet une analyse détaillée de l'algorithme de l'application,
  - homogénéité : disponibilité des logiciels sous différentes plateformes,
  - flexibilité : possibilité d'ajuster l'application aux besoins.

études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source

- Négatif :
  - Manque de compatibilité avec les applications commerciales,
  - faiblesse des services et du support technique,
  - la direction de développement des logiciels open source est peu claire,
  - documentation incomplète,
  - nombre réduit d'applications complètes.

Pour plus d'informations sur la migration de la ville de Gdańsk, voici deux adresses Internet qui nous ont aidées à réaliser ce point :

- Page Internet de l'IDABC concernant Gdańsk : http://europa.eu.int/idabc/en/document/4693/470
- Site de la ville de Gdańsk : http://www.gdansk.pl/en/

Text 16: Gdańsk : liens de référence

Estrémadure (Extremadura)

Estrémadure est la communauté autonome d'Espagne la plus pauvre. Elle a décidé en 2002 d'abandonner Microsoft dans les écoles et les administrations. Elle a alors investi dans les logiciels libres et est à la base de la distribution gnuLinEx ou LinEx. Cette distribution est basée sur la distribution Debian avec le bureau GNOME.

Elle a investi dans cette distribution environ 300.000€. En la mettant en place dans les écoles, elle a pu ainsi épargner 30 millions d'euros.

En installant LinEx dans les écoles et dans les administrations, Estrémadure est considérée comme pionnière de l'open source en Espagne. Sa démarche a d'ailleurs été suivie par l'Andalousie qui participe depuis avril 2003 au développement de LinEx.

LinEx propose des logiciels open source comme OpenOffice, Gftp, Gnumeric, Mozilla.

Les points forts de LinEx sont :

- stabilité,
- facilité d'installation,
- libre distribution et utilisation,
- pratiquement aucun risque par rapport aux virus informatiques,
- coût nul de la licence,
- adaptation à l'espagnol.

Figure 12: Les communautés autonomes d'Espagne

Ses points négatifs sont :

- l'incompatibilité partielle avec certains types de formats propriétaires comme ceux de Microsoft par exemple,
- peu de magasins vendent des ordinateurs avec LinEx pré-installé.
- Le nom de beaucoup de programmes contenus dans la distribution a été changé par d'autres en rapport avec Estrémadure, ce qui peut compliquer son apprentissage et son usage. Dans la version 2004, il a toutefois été inclus la possibilité d'opter pour l'utilisation des noms originaux de toutes les applications incluses dans LinEx. Malgré tout, beaucoup d'utilisateurs voient cela comme une vertu, puisqu'il permet d'appeler ses applications préférées avec des noms communs et sans anglicismes.

De part sa pauvreté, Estrémadure peut ainsi consacrer ses revenus à d'autres missions plus importantes que le payement de licences. Le mouvement open source, et principalement LinEx, fonctionne tellement bien dans cette région d'Espagne que 20% des habitants utiliseraient quotidiennement LinEx chez eux.

Cette expérience nous montre bien que l'open source peut-être mis en place et utilisé dans des environnements variés (administrations, école et privé) si le gouvernement décide d'y accorder une certaine attention. Et que l'utilisation de tels logiciels par le secteur public permet d'épargner des sommes importantes qui peuvent alors êtres consacrées à des tâches de plus grandes importances comme le développement de l'emploi, des infrastructures, etc.

études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source

Pour plus d'informations, nous conseillons les articles et sites Internet suivants, qui nous ont aidés à réaliser ce point :

- Le site officiel de LinEx en espagnol : http://www.linex.org/
- Le site officiel d'Estrémadure en espagnol : http://www.juntaex.es/
- Sweeping initiative puts 80,000 computers running GNOME into student's hands in the region of Extremadura, Spain (http://www.gnome.org/press/releases/extremadura.html, 28/04/2006)
- GNU/LinEx (Free Software), Extremadura Regional Government (http://www.linex.org/linex2/linex/ingles/index_ing.html, 28/04/2006)
- Forget Munich's Linux Migration, It's Already Done by Extremadura, Xavier Calbet (http://www.osnews.com/story.php?news_id=12611, 28/04/2006)
- Wikipedia avec les mots Estrémadure et LinEx qui fournissent d'autres liens intéressants

Text 17: Estrémadure : liens de référence

2.1.2 Nationales

Schoten

Municipalité se situant dans la province d'Anvers. Elle s'étend sur 29,55km², avait 33230 habitants le 1er janvier 2005. Au niveau technique, elle compte une vingtaine de serveurs et environ 200 postes de travail et portables.

Les **motivations** ayant poussée à la migration vers l'open source sont :

- le coût des licences,
- la facilité de configuration,
- l'interoprérabilité,
- le principe de mutualité,
- l'intérêt des citoyens (public interest),
- la stabilité,
- la sécurité,
- le clustering.

Méthode utilisée

Les serveurs ont migré l'un après l'autre. Chaque migration ayant été relativement bien préparée : tests de la solution dans un environnement « fictif » et dans un environnement « réel » et mise en place de scénarios en cas d'échec.

Comme le dit Jan Verlinden, responsable de la migration : « *A good preparation of the migration, testing of the new servers and having a "fallback scenario" at hand, makes every migration to OSS a piece of cake, to my opinion ».*

La migration initiale fut mise en place avec l'aide de consultants externes mais le support quotidien n'en nécessite pas. En effet, les compétences internes sont suffisantes d'après Jan Verlinden.

Niveau de réussite

Jan Verlinden estime que la migration est un franc succès. Elle a permit d'augmenter la productivité, ce qui est confirmé par les employés. Et les avantages suivants ont été dégagés :

- système plus stable,
- système plus adaptable,
- gain de connaissances pour le personnel,
- meilleure utilisation des impôts,
- gain par échange d'informations sur les logiciels avec d'autres,
- meilleure procédures de sauvegarde,
- pas de licences à payer,
- moins d'erreurs « comme » dans les système basés sur Windows.

Suivant l'administration, cette migration permet d'économiser environ 50% du budget informatique. Ils envisagent d'ailleurs d'installer OpenOffice sur les postes de travail.

Pour plus d'informations sur la migration de la ville de Schoten, voici deux adresses Internet qui nous ont aidées à réaliser ce point :

- Page Internet de l'IDABC concernant Schoten : http://europa.eu.int/idabc/en/document/5178/470
- Site de la ville de Schoten : http://www.schoten.be/

Text 18: Schoten : liens de référence

Ciney

Contexte

Entre 2002 et 2003, la ville de Ciney ont reçu un subside, de la Région Wallonne, de 8.000.000 de francs belge pour mettre en place un espace numérique citoyen, c'est-à-dire un local, équipé d'ordinateurs connectés à Internet, accessible aux citoyens de la commune qui le désirent. Cet espace doit proposer, tout au long de l'année, des plages horaires ouvertes au public ainsi que des périodes de formations.

Avec le budget prévu, Monsieur D. Delaunois a étudié la possibilité de mettre en place une

études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source

solutions basée sur des logiciels propriétaires (principalement les produits Microsoft) et une solution basée sur des logiciels open source. C'est la deuxième solution qui fut choisie. En effet, l'utilisation de logiciels open source permettait de créer un plus grand espace numérique citoyen et permettait aussi d'actualiser l'informatique de la commune. Cette différence s'est jouée au niveau financier, avec l'open source, il est possible d'épargner le prix des licences et par là investir cette somme dans les formations et l'achat de matériel informatique complémentaire. C'est ce qui a été fait à Ciney.

Méthode utilisée

Monsieur D. Delaunois ne s'est pas lancé dans la mise en place d'une solution open source sans filet. Il a commencé par faire appel à un prestataire de services qui l'a aidé dans les premières étapes de la migration.

Le plan utilisé pour mettre en place la solution peut-être divisé en différentes étapes :

- Analyse des coûts et choix d'une solution : ici, choix entre une solution propriétaire et une solution open source.
- Analyse de l'existant : tant au niveau matériel, pour vérifier la compatibilité matériel-logiciel, qu'au niveau logiciel, il a ainsi pu récupérer les documents existants, mettre les petites bases de données dans une plus grande générale.
- Phases de tests : les serveurs ont peu été testés car il n'en existait pas vraiment avant, ils ont donc été mis directement en production avec la possibilité si un problème survenait de revenir en arrière sans pertes importantes. Les logiciels bureautiques ont été testés par lui-même dans son travail quotidien et par certains utilisateurs finals volontaires.
- Formations : il a mis en place des séances d'informations et de formations aux nouveaux outils qui allaient être mis en place. Ce qui a permis de limiter les problèmes liés à la peur du changement et à la diminution de la productivité.
- Mise en production : les logiciels utilisateurs ont finalement été mis en production.
- Veille technologique : il teste régulièrement de nouveaux logiciels open source et les met en production s'il les estime utiles, meilleurs que les anciens. Généralement, il met à jour les logiciels déjà utilisés. Par exemple, le passage d'OpenOffice 1 vers OpenOffice 2 est actuellement envisagé.

Niveau de réussite

Au niveau technique Monsieur D. Delaunois est satisfait de ce qui a été mis en place et tout fonctionne pour le mieux. Au niveau de l'utilisation du système, il est aussi satisfait car il a pu éviter le rejet de la solution. Pour se faire, il a beaucoup misé sur l'information et la formation des utilisateurs finaux. Il a surtout présenté la migration via ses aspects positifs en leur montrant les améliorations que les nouveaux logiciels allaient avoir sur leur travail quotidien. Il leur a aussi fait testé différentes solutions pour voir celles qui leur convenaient le mieux. Et surtout, il est toujours resté disponible pour répondre aux questions.

Il faut toutefois admettre que certains services communaux n'ont pas encore migré sous open source. Et ce principalement pour des raisons techniques, souvent les logiciels open source

spécifiques nécessaires au fonctionnement de ces services ne sont pas disponibles ou encore trop immatures pour être utilisés quotidiennement.

Retour d'expériences

Au niveau de l'administration, il n'y a pas eu de réelle étude de satisfaction après la migration.

Mais nous pouvons déjà dire que l'expérience de l'espace citoyen numérique sous logiciel open source est un franc succès. En effet, d'après monsieur X, les 60 ans et plus qui assistent à des séances d'information/formation à l'informatique demandent généralement qu'on leur installe Linux sur leur ordinateur personnel.

« *Les 60 ans et plus sont 85% à demander une installation de Linux après 4 séances* » d'après Monsieur D. Delaunois.

EASI-WAL

Contexte

Au sein du commissariat EASI-WAL, la migration vers une solution bureautique s'est imposée lors de sa création qui consistait à fusionner Wall-On-Line[49] et le CSA[50]. Comme ces deux entités avaient des systèmes informatiques différents, il fut décidé de mettre en place un nouveau système qui reprendrait les données des deux entités.

Il fut alors décidé de mettre en place un système basé principalement sur des logiciels open source. La mise en place des serveurs basés sur Linux n'a posé aucun problème technique. Et la mise en place de stations de travail basées sur Windows XP avec un package de logiciels open source n'a posé aucun souci particulier.

Les logiciels choisis furent :

- OpenOffice à la place de Microsoft Office,
- Firefox à la place d'Internet Explorer,
- Thunderbird à la place d'Outlook Express,
- Phprojekt à la place d'Outlook, de MSProject ou de Teamware.

Les autres raisons de la mise en place de logiciels open source furent :

- l'économie réalisée grâce à l'économie des licences,
- montrer aux administrations que de telles solutions étaient viables.

Méthode utilisée

Il n'y a pas réellement eu de plan de migration pour la mise en place de la solution : Linux pour les serveurs et Windows XP plus logiciels open source pour les stations de travail. Cette solution fut approuvée après avoir été testée aux niveaux fonctionnel et technique.

49 « Wall-On-Line » était le nom donné au projet de Gouvernement électronique de la Région Wallonne.
50 Le CSA = le Commissariat à la simplification administrative.

études de cas d'expériences pertinentes dans le monde administratif de passage à l'open source

Ce qui a demandé plus de temps fut le choix de certains logiciels notamment Phprojekt.

Une fois la solution trouvée et les logiciels choisis, la mise en production fut immédiatement réalisée.

Niveau de réussite

La réussite technique n'est pas à discuter, tout fonctionne correctement.

Le seul bémol concerne Phprojekt qu'il faut adapter aux besoins propres de cette organisation. Cette adaptation est en cours[51].

Au niveau social, tout s'est déroulé dans le calme et les nouveaux outils sont généralement bien acceptés. Bien que quelques problèmes d'utilisation se posent de temps en temps. Cette acceptation est certainement due au nombre important d'informaticiens présents dans l'organisation et au nombre réduit de personnes travaillant dans cette organisation, une vingtaine en tout.

Retour d'expérience

Pour voir ce que les membres d'EASI-WAL pensaient de leurs nouveaux outils de bureautique, nous leur avons fait remplir un questionnaire, celui-ci peut être trouvé dans le troisième point de l'annexe. Nous allons analyser leurs réponses logiciels par logiciels :

- Firefox

  Ce logiciel n'a posé aucun problème d'adaptation pour l'utilisation qui en est faite. Les seuls difficultés rencontrées concernent les sites Internet qui ne sont lisibles qu'avec Internet Explorer.

- Thunderbird

  D'après les membres d'EASI-WAL la prise en main de ce logiciel dans ses fonctionnalités de messagerie fut assez aisée de part sa similarité avec Outlook de Microsoft. Ils regrettent toutefois l'absence des fonctions calendrier, tâches et notes présentes dans ce dernier.

- OpenOffice

  Après quelques mois d'utilisation, ils estiment en général qu'OpenOffice est un logiciel « équivalent » à Microsoft Word pour une majorité de fonctionnalités. Ils rencontrent néanmoins quelques difficultés lors de son utilisation à cause des réflexes acquis après une longue période d'utilisation de Microsoft Word. C'est pourquoi une grande majorité des employés estiment qu'une formation aurait eu sa place pour simplifier leur travail, acquérir de nouveaux automatismes et ne pas perdre leur temps à chercher comment faire.

  Ils rencontrent aussi quelques difficultés lors de l'échange de fichiers avec les administrations, les Cabinets et les consultants. Ces difficultés sont liées aux formats de fichiers différents, OpenDocument d'un coté et Microsoft Office de l'autre, et aussi à la faible diffusion d'OpenOffice auprès de leurs collaborateurs.

51 Cette adaptation fut ma principale activité lors de mon stage au sein du commissariat EASI-WAL.

- Phprojekt

  Il ressort de l'enquête que c'est le logiciel qui pose le plus de problèmes. Il est considéré comme lent, lourd et compliqué à utiliser mais il est aussi fortement apprécié car il permet d'y accéder depuis chez soi.

  Ce logiciel aurait demandé une formation aux fonctionnalités de base pour que tout le monde l'utilise de la même manière. Il doit aussi être adapté aux besoins d'EASI-WAL, c'est-à-dire simplification de certaines fonctionnalités et ajout d'autres.

  Il est difficile de se prononcer sur la viabilité à long terme de ce logiciel. C'est pourquoi, ils restent attentifs à l'évolution des logiciels concurrents.

Nous pouvons donc dire que la migration s'est assez bien déroulée avec quelques petits problèmes d'adaptations. Mais de manière générale chacun apprécie l'idée de travailler avec des logiciels open source. Ce n'est pas eux qui sont parfois remis en cause mais bien leur qualité et surtout leur ergonomie qui diffère des produits Microsoft.

Pour conclure, nous leur conseillons de mettre en place une formation pour le personnel qui utilise de manière quotidienne ces nouveaux outils qui diffèrent de ceux utilisés anciennement.

3 Identification et analyse de guides d'étapes existants

Des études de cas du point précédent, nous pouvons extraire certaines étapes récurrentes réalisées lors d'une migration :

- le choix d'une solution parmi plusieurs possibilités,
- le test des solutions possibles et celle choisie,
- l'analyse des coûts de la migration,
- la formation et l'information des utilisateurs et mainteneurs de la solution choisie,
- la mise en place « progressive » de la nouvelle solution avec des sécurités de retour en arrière si quelque chose ne fonctionne pas correctement,
- l'analyse à posteriori de la réussite de la migration par rapport à son utilisation.

Et nous voyons que la principale motivation, pour la mise en place d'une solution basée sur des logiciels open source, se situe au niveau financier avant d'être technique ou philosophique

Nous allons maintenant grâce à différents guides d'étapes existants voir si ces étapes sont pertinentes et proposer des étapes qui seront détaillées dans le chapitre suivant.

3.1 Guides d'étapes

Les différents guides d'étapes à être présentés seront :

- Le guide IDA de migration vers l'Open Source, qui peut-être considéré comme une référence,
- Managing Linux Migration, un article simple et qui fait les grandes lignes d'étapes,

Identification et analyse de guides d'étapes existants

- Migration vers OpenOffice.org sous environnement Windows : analyse d'expériences internationales.

Il est évidemment possible de trouver beaucoup d'autres documents relatifs aux étapes de migrations. Ces documents sont souvent focalisés sur une migration particulière comme comment passer de Windows 2000 à Windows XP. Ces documents peuvent toutefois être instructifs pour toute migration envisagée car ils reprennent généralement les grandes étapes d'une migration appliquées à un exemple concret.

3.1.1 Le guide IDA de migration vers l'Open Source, [Choppy, 2003]

Le guide IDA de migration vers l'Open Source est une référence proposée par la Communauté européenne.

Ce guide propose dès le début une méthodologie à suivre pour réaliser une migration :

« Tout exercice de migration consiste généralement en :

1. une phase de collecte d'informations et de définition de projet comportant :

   A) une description de l'ensemble des conditions initiales consistant, par exemple, en :

      a) architecture(s) système,

      b) applications et leurs données associées,

      c) protocoles et standards utilisés,

      d) matériel,

      e) environnement physique tel que bande passante réseau et localisation,

      f) pré-requis sociaux tels que les langues et ensembles de particularités des équipes ;

   B) un ensemble de conditions cibles du même niveau de détail,

   C) une description de la démarche permettant le passage des conditions existantes à celles planifiées ;
2. une justification de la migration incluant le calcul des coûts associés ;
3. une ou plusieurs phases pilotes conçues pour tester le plan et la justification. Les données issues de ces pilotes peuvent ensuite être réinjectées dans le modèle de coûts utilisé dans le plan ;
4. déroulement du plan ;
5. suivi et actualisation de l'expérience réelle acquise par rapport au plan. »

Il propose aussi un processus de migration qui mérite notre attention, voir le quatrième point de l'annexe.

Il nous apprend aussi :

- qu'il est parfois nécessaire de modifier l'environnement existant avant d'envisager une migration vers une solution open source.

- qu'il est nécessaire de prendre en compte les critères humains (peur de l'inconnu, effet CV[52], connaissance = pouvoir, etc.).
- qu'il existe des méthodes pour faciliter les choses :
    - introduire les nouvelles applications dans un environnement familier,
    - commencer par le plus simple,
    - penser à plus loin, c'est-à-dire limiter la complexification pour faciliter les migrations ultérieures.

Dans la partie plus technique, il propose une solution envisageable en analysant les logiciels les plus connus poste par poste et les problèmes qu'il pourrait rencontrer comme par exemple la réception de fichiers non lisibles par une application de traitement de texte.

Dans cette partie technique, il propose aussi une série de scénarios de migrations sur les points techniques importants à prendre en compte dans tel ou tel cas. Le scénario le plus important et le plus documenté concerne le cas le plus fréquent c'est-à-dire le passage d'un système basé sur Windows à un système basé sur Linux.

En annexe, il propose une liste non exhaustive de logiciels open source tant pour les postes de travail que pour les serveurs.

3.1.2 Managing Linux Migration, [Deb & Sirvastava, 2004]

Cet article nous propose certains avantages, principalement financiers, à passer sous une solution basée sur Linux. Il explique qu'il est important de comparer les solutions Windows et Linux avant de faire un choix définitif car il existe beaucoup de sociétés qui ont réalisé d'importantes économies en migrant vers Linux.

Il explique aussi que le management des risques est plus qu'important dans une migration, c'est pourquoi, il conseille de commencer par des services non critiques comme certains serveurs Internet, de fichiers ou d'impression.

Il propose de se focaliser sur les cinq étapes suivantes, étapes reprises dans la Figure 13 :

- Experiment : Dans cette étape, on envisage de tester la solution sur des serveurs non critiques comme ceux cités ci-dessus. Cette étape devrait montrer après quelques mois les avantages de Linux. Cette étape permet aussi de se familiariser avec cette solution.
- Pilot : Cette étape consiste à tester des applications intensives, non critiques et non transactionnelles dans un premier temps. Une fois que ces applications ont donné satisfaction, on peut envisager de tester des applications critiques et transactionnelles.
- Test : Une fois que les tests ont été concluants, on peut mettre en production les applications citées à l'étape précédente.
- Consolidation : La réussite de la migration des applications migrées dans les étapes précédentes devrait influencer positivement les décideurs à migrer le reste de leurs

52 « Effet de dilution de CV : Aussi bien l'équipe système que les utilisateurs peuvent penser que la non-utilisation de logiciels standard risque d'altérer leur capacité de développement de carrière. » [Choppy, 2003] page 20.

applications sous Linux. Cette migration devra se faire suivant un plan précis et complet.

- Continuous adoption : Les nouvelles applications devront être mises sous Linux et les applications centrales sont sous Linux.

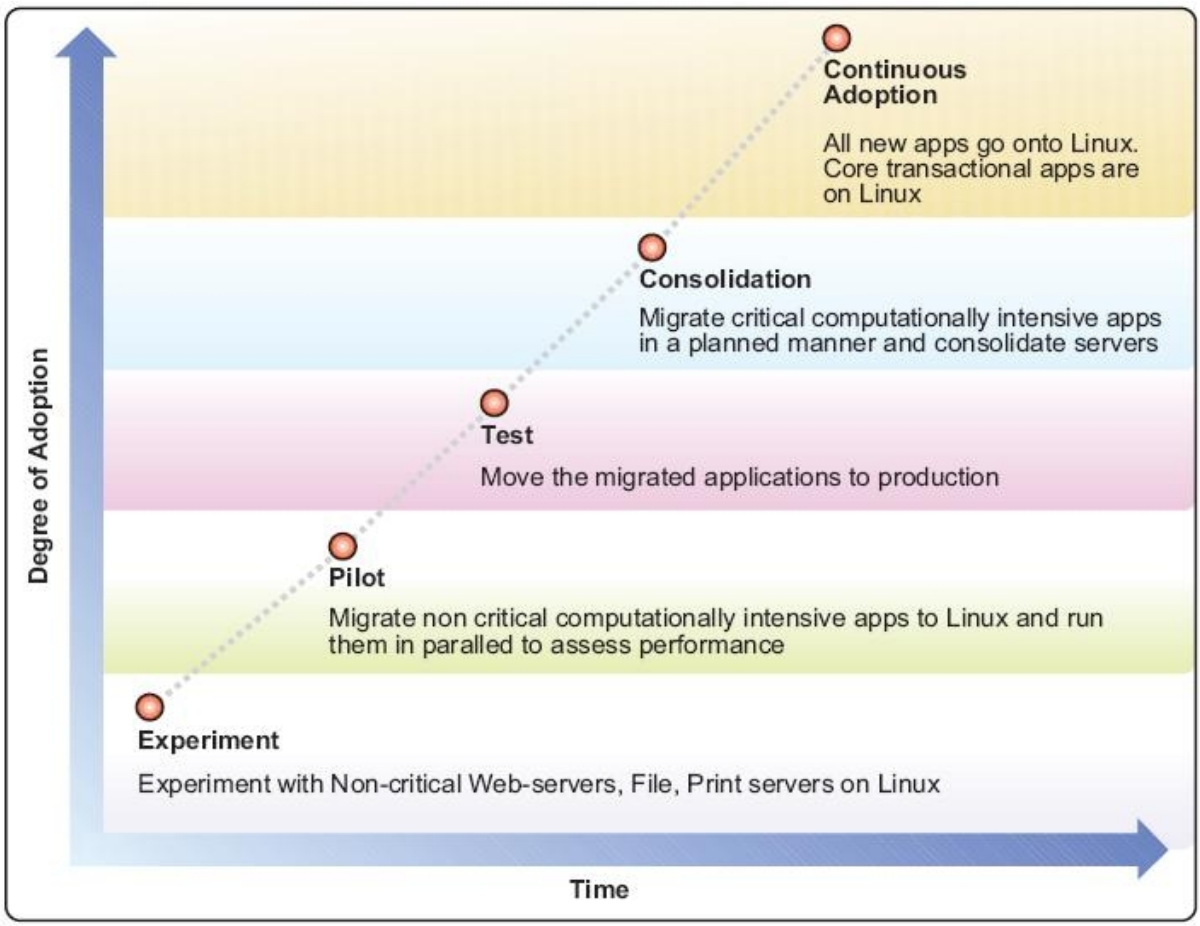


Figure 13: Infosys Linux Adoption Roadmap

3.1.3 Migration vers OpenOffice.org sous Environnement Windows : analyse d'expériences internationales, [Aboubekr & Rivard, 2005]

Ce document ne reprend pas un plan de migration en temps que tel mais insiste sur différents points qui sont importants à prendre en compte dans différentes étapes d'une migration. Ce document présente trois cas de migrations vers OpenOffice. Ces cas sont : le cas des mairies de Vaulx-en-Velin et de Vessinieux, le cas de l'hôpital Avranche-Granville et le cas de la Gendarmerie nationale française.

Ce document reprend deux points principaux, le premier nommé « la constellation de facteurs clé » regroupe une série de facteurs importants pour réussir une migration. Le deuxième point important est l'analyse des coûts.

La constellation de facteurs clé regroupe :

- La vision : peut-être assimilée au fait de connaître l'état futur désiré et de savoir comment y accéder. Cette vision doit être diffusée et acceptée par un grand nombre pour que la migration se passe sans problèmes majeurs.
- Les processus de gestion :
  - la gestion du changement : dépendra du type de changement que le projet va imposer à l'organisation.
  - la gestion de risque : doit toujours être prise en compte.
  - la gestion contractuelle : lors d'une migration, on choisit des produits mais il faut aussi faire attention aux fournisseurs que nous allons choisir. Il faudra donc se poser des questions comme « Sont-ils fiables? Ont-ils une bonne réputations? Etc ».
- Les habilités de gestion :
  - gagner l'adhésion : de toute l'organisation aux changement qui vont avoir lieu est un plus pour la réaliser sans soucis sociaux.
  - le champion : est une personne qui fait la promotion de la vision et permettra à cette dernière de remporter un franc succès.
- La structure d'une équipe de projet est importante pour la réussite de ce projet. Il faut donc la choisir avec soin et bien définir le rôle de chacun.

L'analyse des coûts est un des facteurs clé qui doit déterminer si une solution est financièrement plus intéressante qu'une autre. Il conseille d'utiliser le TCO[53] (Total Cost of Ownership) qui permet généralement d'analyser les solutions possibles au niveau économique et ce sur une période suffisamment longue.

3.2 Propositions d'étapes

Les différentes étapes suivantes, définies dans le chapitre suivant, ont été déterminée grâce à l'analyse des études de cas et des guides d'étapes ci-dessus :

1. Étapes préliminaires :
   a) Lancement de la migration,
   b) Définition des équipes chargées de la migrations,
2. Analyse de l'existant matériel,
3. Analyse de l'existant logiciel,
4. Proposition de solutions candidates,
5. Analyse technique et fonctionnelle,
6. Analyse économique,

53 Pour plus d'information, se reporter au Text 21 au chapitre 6, page 103.

Identification et analyse de guides d’étapes existants

7. Choix de la solution finale,
8. Présentation de la nouvelle solution,
9. Formation de l'équipe IT,
10. Tests d'intégration de la nouvelle solution,
11. Formation des utilisateurs,
12. Mise en place de la nouvelle solution.

Il est évident qu'en fonction du type de migration envisagé, l'importance de chaque étape sera différente. Cependant, nous conseillons de prendre en compte toutes ces étapes quelque soit le type de migration envisagé.

Chapitre 6 : Rédaction du guide d'étapes sur base du travail préparatoire

Ce chapitre va développer les différentes étapes proposées précédemment.

Nous pouvons les comparer à celles qui vont du lancement d'un cahier de charges à la mise en place de celui-ci.

En effet, la première étape va nous permettre de déterminer si la migration est nécessaire ou non et va mettre en place les différents comités qui seront chargés de la suivre tout au long de son évolution. C'est comparable à la nécessité ou non de réaliser un cahier de charges. Et si tel est le cas, la mise en place d'équipes chargées de le rédiger et de déterminer, par la suite, la proposition la plus adéquate doit être envisagé.

Les deuxième et troisième étapes, l'analyse de l'existant matériel et logiciel, doivent nous permettre de fournir les exigences du nouveau système, de la même manière que nous déterminerions les exigences liées à un cahier de charges. En effet, ces deux étapes nous fournissent toutes les informations nécessaires pour connaître les fonctionnalités importantes du système actuel que nous devrons rencontrer dans le système à mettre en place. Il est possible et même conseillé de réaliser ces deux étapes en parallèle.

La quatrième étape, proposition de solutions candidates, est comparable au dépôt de réponses au cahier de charges et au premier tri de ces dernières.

Les étapes cinq et six, analyse technique et fonctionnelle et analyse économique, ont comme objectifs :

- premièrement, éliminer les solutions les moins intéressantes, de la même façon que nous éliminerions les réponses au cahier de charges qui sont peu convaincantes dès le départ.
- et deuxièmement, classer les différentes solutions restantes sur base de leur coût et de leur capacité à répondre aux besoins exprimés à la suite de l'analyse de l'existant matériel et logiciel. C'est comme si nous classions les réponses au cahier de charges restantes grâce à leur pertinence.

Un fois ces deux objectifs atteints, nous avons, normalement, suffisamment d'informations pour passer à la septième étape qui consiste à choisir une solution parmi celles qui ont été

sélectionnées. Comme nous devrions choisir la proposition de solution au cahier de charges qui remplit au mieux nos besoins.

Finalement, les cinq dernières étapes, présentation de la nouvelle solution, formation de l'équipe IT, tests d'intégration de la nouvelle solution, formation des utilisateurs et mise en place de la nouvelle solution, ont pour principal objectif de permettre la mise en production de la solution choisie dans les meilleures conditions possibles. Ces étapes sont semblables à celles qui nous amèneraient à la mise en place de la solution au cahier de charges sélectionnée.

1 Étapes préliminaires

1.1 Lancement de la migration

Avant de se lancer dans les étapes qui suivent, il est intéressant de se poser la question : « Pourquoi allons-nous réaliser une migration et quelle est sont but ? ». La réponse à cette question devrait être claire et précise. Si ce n'est pas le cas, il serait intéressant que nous redéfinissions les raisons qui poussent à réaliser cette migration et peut être constaterons nous qu'elle n'a pas encore lieu d'être.

Cette première étape préliminaire a donc comme objectif de déterminer si lancer une procédure de migration est pertinent ou pas. Ce sont les responsables de l'infrastructure IT ainsi que les cadres dirigeants de l'organisation qui doivent prendre cette décision.

Il est aussi important de poser l'étendue de la migration. En effet, en fonction de cette étendue, certaines étapes proposées ci-dessous n'auront pas le même poids que ce qui sera présenté car nous nous basons ici, sur une migration importante et rapide.

Par exemple, réaliser une refonte totale du système informatique, changer le navigateur Internet ou le serveur mail ne demandent pas les mêmes ressources.

1.2 Définition des équipes chargées de la migration

Lors d'une migration, nous avons besoins de déterminer les responsabilités de chacun et de mettre en place les équipes chargées de la migration et de la prise de décisions. Ces deux équipes peuvent comprendre des membres communs mais auront aussi des tâches spécialisées.

La première équipe sera amenée à gérer la migration. Cette équipe doit pouvoir suivre l'évolution de la migration. Cette équipe devrait être composée de responsables de l'organisation, de participants à l'étude de faisabilité de la migration, d'utilisateurs qui seront touchés par la migration et de personnes qui seront responsable de son implantation dans l'organisation. Dans cette équipe, nous pouvons aussi retrouver des experts indépendants qui ont parfois une vision plus globale de la situation.

La seconde équipe est celle qui sera amenée à choisir la solution finale. Elle sera donc composée de dirigeants de l'organisation et de membres de l'équipe de gestion de la migration.

2 Analyse de l'existant matériel

2.1 Définition et objectifs

Cette étape consiste à répertorier les différents matériels utilisés. Elle aura pour objectif la création d'une base de données objective et exhaustive de tout le matériel informatique utilisé par l'organisation qui envisage de changer d'environnement.

Nous considérons cette étape comme cruciale et pouvant se faire en parallèle avec l'analyse de l'existant logiciel. En effet, il est capital de savoir ce que nous possédons, cela permet, lors de l'analyse des solutions candidates, de savoir ce qu'il faudra changer comme matériel et le coût que cela impliquera.

Normalement, si une politique de gestion informatique forte existe, cette étape ne devrait pas prendre beaucoup de temps. Si par contre l'organisation n'a pas de politique de gestion du matériel, cette étape prendra un certain temps. Et nous constaterons l'intérêt d'une politique de gestion du matériel informatique forte et suivie pour le futur.

2.2 Contexte d'utilisation de la méthode

Il est évident que si nous envisageons, dès le lancement du projet, de changer tout le matériel informatique, cette étape ne sera pas utile. Comme généralement ce n'est pas le cas et que les gestionnaires poussent à éviter le plus possible de changer du matériel qui coûte cher, cette étape sera cruciale pour cibler les modifications et les achats à réaliser en maximisant la valeur de l'existant.

2.3 Les acteurs

2.3.1 Qui participe à l'analyse de l'existant matériel ?

Principalement les informaticiens avec l'aide et la participation passive, en général, des utilisateurs. En effet, dans certains cas, il n'est pas nécessaire de déranger les utilisateurs pour savoir ce qui est utilisé comme matériel. Dans d'autres cas, il faudra passer sur chaque machine et faire une analyse précise de ce qu'elles contiennent (processeur, carte graphique, Ram, etc.).

2.3.2 Comment recruter et convier les participants ?

Pour la participation des informaticiens, il ne faut rien faire de particulier car cette activité peut entrer entièrement dans leur compétences.

Pour la participation des utilisateurs, il suffira généralement de leur envoyer une note leur expliquant la raison du passage des informaticiens et leur expliquer, dans les grandes lignes, ce qu'ils feront sur leur ordinateur.

2.4 Mise en oeuvre de la méthode

2.4.1 Quel est la méthode à utiliser pour réaliser une analyse de l'existant matériel ?

Il existe plusieurs méthode, la première consiste à utiliser des logiciels qui automatisent la recherche des informations.

La deuxième méthode consiste à faire un inventaire à priori basé sur les informations disponibles. Ces informations peuvent être récupérées via la comptabilité, connaissance des achats de matériel, et via le service informatique, connaissance des placements de matériel effectué.

La troisième méthode est une méthode plus pragmatique, simple mais probablement un peu plus longue. Cette méthode demande une certaine préparation pour les informaticiens et peut être caractérisée par les étapes suivantes :

- Mise au point d'une mini application de recensement des données.

  Une mini application web qui permet d'encoder les différents composants clé de chaque ordinateur. Application qui envoie et stocke les données dans une base de données relationnelle.
- Prévenir les utilisateurs que les informaticiens vont passer dans leur bureau et leur expliquer la raison de ce passage.

  Par exemple en envoyant un mail à tous les utilisateurs.
- Encoder les données.

  En encodant directement via la mini application web développée.
- Analyser les données récoltées.

  Référencer les différentes configurations qui existent dans le groupe. Et pour chaque configuration établir le nombre d'ordinateurs qui y correspond.

Ces méthodes seront d'autant plus intéressantes et efficaces si elles sont basées sur les bonnes pratiques mises en place par ITIL[54] et si elles sont liées à une CMBD[55].

Il est évident que ces trois méthodes peuvent être utilisées de concert pour obtenir un résultat optimal.

2.4.2 Quelle est la durée de l'analyse de l'existant Matériel ?

Pour les deux premières méthodes, le temps dépendra principalement de la taille de l'organisation et de sa répartition géographique.

Pour la troisième méthode, chacune des étapes proposée utilisera un temps différent qu'il

54 ITIL, Information Technology Infrastructure Library, est un ensemble de bonnes pratiques pour la gestion d'un système d'information, édictées par l'Office public britannique du Commerce. Elle reprend notamment un livre consacré à la gestion des infrastructures des TIC. (ref : Wikipedia)

55 La CMBD, Configuration Management DataBase, est une base de données unifiant les composants d'un système informatique. Elle permet de comprendre l'organisation entre ceux-ci et de modifier leur configuration. La CMDB est un composant fondamental d'une architecture ITIL. (ref : Wikipedia)

faudra cumuler et qui dépendra principalement du nombre de visites chez les utilisateurs qu'il faudra réaliser.

- La première étape ne devrait pas occuper plus d'une personne pendant une journée. Et si la personne est habituée à ce genre de travail, deux ou trois heures peuvent être suffisantes.
- La deuxième étape ne devrait pas prendre beaucoup de temps. Temps nécessaire : rédiger le mail et l'envoyer aux personnes devant faire l'objet d'une visite.
- La troisième étape sera la plus longue et dépendra du nombre d'informaticiens réquisitionnés pour la réaliser et le nombre d'utilisateurs à visiter.
- Le temps nécessaire pour la dernière étape peut être court si les besoins initiaux sont limités. Ce qui est normalement le but de cette activité.

2.4.3 Quel est le résultat de l'analyse de l'existant matériel ?

Le résultat attendu est une base de données complète et exhaustive de tout le matériel informatique utilisé par l'organisation envisageant le changement de système.

2.5 Atouts et limites

2.5.1 Atouts

Si cette étape est bien réalisée, elle entraîne une connaissance précise de ce que le groupe possède comme matériel informatique. Et elle permettra une meilleure gestion future du parc informatique si la gestion informatique met à jour les données à chaque changement de matériel.

Elle permettra lors de l'analyse des différentes solutions d'évaluer précisément le matériel compatible et donc de savoir ce qu'il faudra changer pour le rendre compatible avec la nouvelle solution qui sera choisie.

Grâce à cette étape, nous limitons donc les risques financiers liés à la compatibilité logiciel/matériel. En effet, on pourra calculer précisément ce que chaque solution envisagée pourrait nous coûter en matériel.

2.5.2 Limites

Il est possible que l'on passe à coté de certains matériels. Matériel non présent lors de l'analyse (non connecté au réseau), matériel oublié, matériel perdu mais retrouvé par la suite sans que le service informatique ne soit prévenu.

Dans le cas de la troisième méthode, il y a la possibilité que certains utilisateurs refusent de participer au recensement.

2.5.3 Quand opter pour cette méthode ?

Lors de chaque migration si l'organisation ne possède pas un outil de gestion en continu de l'évolution de son parc informatique.

2.6 Informations complémentaires

- [Oettinger, 1999-2001]
- Exemple de logiciels aidant à évaluer son parc informatique :
  - Outils Microsoft : http://www.microsoft.com/france/logicieloriginal/gerer/audit/outils.mspx,
  - Synexsys Inventory de Data Concept : http://www.dataconcept.ch/synexsys/index.htm,
  - i.Parc de Logimot : http://www.logimot.com/sitefr/iparc.htm,
  - Isiscan de Isilog : http://www.isilog.fr/Helpdesk-Inventaires-Parcs-Infrastructures/Inventaire-Informatique-Par-Le-Web/Logiciel-d'inventaire-de-parc-informatique.asp,
  - d'autres exemples de logiciels sur le site de ZDNet : http://www.zdnet.fr/produits/materiels/pc/0,39049784,39213652,00.htm.

3 Analyse de l'existant logiciel

3.1 Définition et objectifs

Cette étape consiste à répertorier les différents logiciels utilisés. Elle aura pour objectif la création d'une base de données objective et exhaustive de tous les logiciels utilisés par l'organisation qui envisage de changer d'environnement.

Cette étape est cruciale et peut se faire en parallèle avec l'analyse de l'existant Matériel. En effet, il est crucial de savoir quels sont les logiciels utilisés par l'organisation et qui les utilisent, nous nous intéressons plus aux postes qu'aux personnes physiques, cela permettra lors de l'analyse des nouvelles solutions candidates, de savoir quels logiciels il faut impérativement avoir et ce qui doit être fait comme changement.

Par exemple, les macros Microsoft Office sont fréquemment utilisés par certaines personnes mais ils sont, actuellement, incompatibles avec OpenOffice. Ce qui demande un travail non négligeable lors d'une migration de Microsoft Office vers OpenOffice.

L'objectif est donc d'avoir une vue globale et précise des logiciels qui sont utilisés au sein de l'organisation et aussi de savoir comment ils sont utilisés. Cette vue globale peut permettre la mise en place d'une standardisation des moyens utilisés. Comme par exemple à Ciney où lors de la migration les bases de données personnelles ont été intégrées dans une base de données générique.

3.2 Contexte d'utilisation de la méthode

Lors de toute migration, il est nécessaire de passer par cette étape même si elle ne cible qu'une catégorie de logiciels bien précise. En effet, il est important de savoir ce qui est utilisé, et comment, pour pouvoir proposer une solution au moins équivalente à celle existante.

Il est probable que, dans les grandes organisations (sociétés nationales ou multinationales), cette étape soit moins importante que dans d'autres plus petites (PME, Asbl, administrations, etc). En effet ces grandes organisations ont généralement des règles d'utilisations

standardisées.

Cette étape est donc primordiale pour les organisations qui ne connaissent pas suffisamment leur système informatique.

3.3 Les acteurs

3.3.1 Qui participe à l'analyse de l'existant Logiciel ?

Les informaticiens doivent participer à cette analyse ainsi que les utilisateurs qui sont les mieux placés pour nous informer sur les logiciels qu'ils utilisent et surtout comment ils les emploient.

3.3.2 Comment recruter et convier les participants ?

En ce qui concerne les informaticiens, cela fait partie intégrante de leur mission lorsqu'une migration est envisagée.

Pour ce qui est des utilisateurs, il est nécessaire d'avoir leur soutien et leur participation. Participation qui sera plus active que lors de l'analyse de l'existant Matériel. Il sera donc nécessaire de leur expliquer clairement ce qu'on attend d'eux et pourquoi. Ici, on ne leur annonce pas les solutions qui pourraient être envisagées mais on leur signale que le système va probablement évoluer, qu'on a besoin de leur aide et qu'ils doivent répondre le mieux possible pour que la nouvelle solution satisfasse à leurs besoins.

3.4 Mise en oeuvre de la méthode

Pour le recensement des logiciels, il existe des logiciels qui automatisent la tâche. Mais nous pouvons aussi nous baser sur l'envoi d'un questionnaire aux utilisateurs. Ce questionnaire sera plus centré sur la manière dont ils utilisent les logiciels que sur les logiciels utilisés.

L'utilisation du questionnaire peut se faire suivant les étapes présentées ci-dessous :

- Rédaction d'un questionnaire standard

  Rédiger un questionnaire standard pour chaque application générique (suite bureautique, groupware, etc.) utilisée et un questionnaire plus ouvert pour les logiciels plus pointus (ex : logiciels de programmation utilisés par les informaticiens, logiciels graphiques utilisés par les graphistes, etc.).

- Les questions porteront principalement sur les fonctionnalités les plus utilisées, et sur celles qui posent problèmes et qui sont indispensables à l'utilisateur.

  Envoi des questionnaire à toutes les personnes travaillant au sein de l'entreprise avec obligation d'y répondre avant une date butoir.

  Distribution sous format papier ou envoi par mail des questionnaires à tous les employés. Ce questionnaire doit être accompagné d'explications sur le pourquoi de ce questionnaire et comment y répondre. Ces explications peuvent se faire soit via informations papiers soit via des réunions. Il est important que l'utilisateur se sente impliqué pour qu'il réponde correctement au questionnaire.

- Réception des réponses et leur analyse.

  En fonction de la méthode utilisée, réception des réponses sous format papier ou numérique. L'analyse devra être réalisée par les personnes ayant créé le questionnaire car elles savent exactement ce qu'elles attendent de ce questionnaire et elles sauront comment classer au mieux les réponses. Si une réponse ne leur convient pas, elles seront les plus aptes à demander des informations complémentaires au répondant.

Si un questionnaire a été mis au point pour l'analyse de l'existant matériel, il peut être intéressant les deux questionnaires soient mis en commun.

Il est nécessaire que les informaticiens à la base du questionnaire soient référencés et présents pour répondre aux questions des utilisateurs qui ne manqueront pas.

Il est recommandé de se baser sur des normes comme celles proposées par ITIL. En effet, ces normes ont déjà fait leur preuves et nous éviterons bien des soucis. Et si le recensement des logiciels est lié au même CMBD que celui de l'analyse de l'existant Matériel, nous aurons une représentation très précise de l'architecture du système informatique de l'organisation.

3.4.1 Quelle est la durée de l'analyse de l'existant logiciel ?

Si nous réalisons un recensement automatisé, la durée de cette étape devrait être principalement liée à l'existence d'une standardisation interne des logiciels et de la diversité des logiciels utilisés.

Si en plus de ce recensement automatique, nous optons pour l'envoi de questionnaire, la durée de cette étape dépendra du temps mis pour rédiger le questionnaire, du temps donné aux utilisateurs pour y répondre et de l'analyse des réponses qui en temps de travail sera l'étape la plus lourde.

Comme nous le voyons, la durée de cette étape dépendra de la profondeur de l'analyse désirée et de l'existence de standardisation interne au niveau des logiciels.

3.4.2 Quel est le résultat de l'analyse de l'existant logiciel ?

L'analyse de l'existant Logiciel doit fournir le nom de chaque logiciel utilisé, pour quels postes ils sont utilisés et comment ils sont employés. Cela devrait donc nous permettre de classer chaque logiciel en fonction de son importance au sein de l'organisation et de connaître pour chaque logiciel les fonctionnalités critiques et celles qui ne le sont pas.

Cela nous permettra lors de l'analyse des solutions de n'oublier aucune catégorie de logiciels et de faire appel aux utilisateurs experts pour qu'ils donnent leur avis sur les solutions envisagées.

3.5 Atouts et limites

3.5.1 Atouts

Impliquer les utilisateurs dans les changements est primordial pour qu'ils soient acceptés, même si la solution finale qui sera choisie dans une étape ultérieure ne pourra pas plaire à tout le monde. Les utilisateurs auront ainsi eu la possibilité de s'exprimer et d'informer les décideurs sur leurs besoins.

Cette étape devrait nous empêcher d'oublier un logiciel vital à l'entreprise et de se retrouver avec une solution non viable. Par exemple, si une application de comptabilité est fortement liée aux formats de Microsoft Excel, il ne faudra pas oublier d'adapter l'application de comptabilité au formats fournis par Calc si on envisage la migration Microsoft Office vers OpenOffice

3.5.2 Limites

Dans certains cas, il est possible de constater que la migration vers une solution réalisable pour la majorité des logiciels est impossible pour un logiciel spécialisé. Dans ce cas, il peut être intéressant d'attendre une solution avant d'aller plus loin dans la migration et de la mettre momentanément en suspens pour certains utilisateurs ou pour tous. Par exemple, à Ciney, le service de cartographie n'a pas encore migré sous Linux car il n'y avait pas, au moment de la migration, de logiciels sous Linux qui correspondaient à leurs besoins.

Il est possible d'oublier un logiciel utilisé de manière exceptionnelle par l'une ou l'autre personne. La solution à ce genre de situation sera à apporter au cas par cas. Si les objectifs de cette analyse ont été clairement expliqués aux utilisateurs et qu'ils participent de manière efficace, ce genre de problèmes sera minimisé.

3.5.3 Quand opter pour cette méthode ?

Dans chaque cas de migration, que nous envisagions de changer la suite bureautique ou de changer radicalement tout le système informatique, une analyse des besoins logiciels est primordiale pour trouver la meilleure solution possible pour les utilisateurs.

Cette analyse de l'existant Logiciel sera d'autant plus importante si la connaissance de l'architecture logicielle du système informatique est pauvre.

3.6 Informations complémentaires

- [Oettinger, 1999-2001]
- Exemple de logiciels aidant à évaluer son parc informatique, voir le point 2.6 Informations complémentaires.
- Des exemples de questionnaires peuvent être trouvés dans [Gonzalez, 2005].

4 Proposition de solutions candidates

4.1 Définition et objectifs

Dans cette étape, nous allons définir différentes solutions envisageables pour remplacer l'ancien système informatique du groupe. Cela signifie que nous allons proposer des logiciels qui peuvent remplacer de façon avantageuse ceux de l'ancienne solution.

Nous suggérons donc pour chaque logiciel existant de trouver les logiciels qui peuvent le remplacer avantageusement. Le choix final d'une solution devra alors se faire après l'analyse technique et économique de chaque possibilité.

Pour bien visualiser l'idée de cette étape, prenons un exemple :

La situation actuelle = Windows 2000 + Microsoft Office 97.

Les solutions possibles sont :

- Windows 2000 + Microsoft Office XP
- Windows XP + Microsoft Office XP
- Windows 2000 + Open Office 2.0
- Windows XP + Open Office 2.0
- Linux + Open Office 2.0
- etc.

Text 19: Exemple de proposition de solutions candidates

4.2 Contexte d'utilisation de la méthode

Cette méthode est importante pour visualiser les solutions candidates.

4.3 Les acteurs

4.3.1 Qui participe à la proposition de solutions candidates ?

Toutes les personnes faisant partie de l'organisation peuvent participer à cette méthode. En effet, les informaticiens ont les plus grandes capacités pour proposer des solutions mais les utilisateurs peuvent aussi apporter des points de vue et des suggestions auxquels les informaticiens n'avaient pas pensés.

4.3.2 Comment recruter et convier les participants ?

Le recrutement des informaticiens va de soi. En effet, cette activité rentre dans leur centre de compétences.

Le recrutement des utilisateurs se fait de lui-même s'ils sont informés qu'on recherche de nouvelles solutions. Ceux qui ont des solutions ou qui veulent en savoir davantage feront les démarches nécessaires pour contacter la cellule responsable du choix des solutions.

4.4 Mise en oeuvre de la méthode

La méthode que nous conseillons d'employer est la suivante :

- Pour chaque logiciel référencé lors de l'analyse de l'existant logiciel devant être remplacé, il faut trouver des logiciels qui pourraient le remplacer de manière efficace

  Ici, il s'agit de sélectionner des logiciels et pas les tester. Dans ce point, nous sélectionnons de manière très large des logiciels susceptibles d'être intéressants mais nous n'allons pas plus loin.

- Tester de manière superficielle chaque logiciel envisagé dans le point précédent. Cela permettra de faire le tri entre les logiciels envisagés respectant les contraintes principales et ceux qui ne les respectent pas.

  Une fois tous les logiciels trouvés grâce au point précédent, il faut tester chaque logiciel de manière superficielle. L'idée de superficialité signifie, tester l'ergonomie générale, voir si les fonctionnalités importantes de la classe de logiciels sont présents. Et s'il peut rencontrer à moyen terme les besoins des utilisateurs. Il ne faut pas avoir peur de demander l'avis des utilisateurs de cette future solution possible. Car c'est ici, que nous devons réaliser le premier tri.
- Fournir une liste des logiciels qui sont envisageables pour remplacer les anciens et définir pourquoi on peut les envisager.

  À partir des logiciels sélectionnés dans le point précédent, fournir une liste des différentes configurations génériques que nous pouvons envisager ou un tableau reprenant les anciens logiciels et les remplaçants possibles. Pour chaque logiciel envisagé, il s'agit de fournir une fiche technique qui décrit ses caractéristiques principales :
    - Nom complet du logiciel
    - Éditeur du logiciel
    - Prix du logiciel
    - Version du logiciel
    - Système d'exploitation sur lequel le logiciel fonctionne
    - etc.

Tableau de solutions candidates

Type de logiciel	***Ancienne Solution***	***Solution 1***	***Solution 2***	***Solution 3***	***...***
Système d'exploitation	Windows 2000	Windows 2000	Windows XP	Linux	
Suite bureautique	Microsoft Office 97	Microsoft Office XP	Open Office 2.0		
Client Mail	Outlook 97	Outlook XP	Thunderbird	Evolution	
Navigateur Internet	Internet Explorer	Internet Explorer	Firefox		
...	...	...	...	...	...

Text 20: Exemple d'un tableau de solutions candidates

4.4.1 Faut-il faire appel à un sous-traitant pour réaliser la proposition de solutions candidates ?

Faire appel à un sous-traitant peut-être utile si les compétences internes sont limitées. Dans ce cas, faire appel à un expert dans ce domaine peut apporter un soutien non négligeable.

4.4.2 Quelles sont les ressources nécessaires ?

La principale ressource nécessaire est le temps fourni par les informaticiens employés pour réaliser cette méthode. Un accès à internet et des machines de tests sont généralement requises pour faire des tests préliminaires des logiciels envisagés.

En effet, il est utile de tester les logiciels envisagés lors de cette étape pour pouvoir éliminer ceux qui ne rencontrent pas les contraintes de base. Proposer des logiciels est facile mais proposer des logiciels performants demande de les tester un minimum. Il est évident que ce n'est pas dans cette étape que l'on va tester en profondeur les logiciels mais bien dans l'analyse technique. Ici, le but est de réaliser le premier tri.

4.4.3 Combien de participants faut-il prévoir ?

Le nombre de participants est fort aléatoire. En effet, si la migration est réduite à un logiciel, le nombre de personnes nécessaires pour établir les solutions ne sera pas élevé. Par contre, si on envisage de modifier le système dans son intégralité (serveurs, stations de travail, portables) le nombre de personnes susceptibles de travailler pour cette étape sera plus nombreux et dépendra de leurs compétences propres.

4.4.4 Quelle est la durée de la proposition de solutions candidates ?

Comme pour le nombre de participants, si on envisage de ne changer qu'un logiciel, il ne faudra peut-être qu'une journée. Par contre pour un système complet, il faudra un peu plus de temps. Mais en moyenne, le temps nécessaire ne devrait pas être excessif et dépendra du nombre de personnes affectées à cet étape et du nombre de logiciels à changer.

4.4.5 Quel est le résultat de la proposition de solutions candidates ?

Comme expliqué plus haut, une liste complète des logiciels qui peuvent être envisagés pour remplacer l'existant et leurs caractéristiques principales.

4.5 Atouts et limites

4.5.1 Atouts

Ne pas limiter ses choix uniquement à des logiciels propriétaires ou open source mais mettre les deux en concurrence. Vu qu'ici, on ne fait pas de tests approfondis, on peut proposer des logiciels qui devront peut être faire l'objet de modifications pour répondre aux besoins internes.

4.5.2 Limites

Certaines lacunes de certains logiciels ne peuvent pas être détectées dans les tests de cette étape. Il est donc possible que de « mauvais » logiciels passent mais, normalement, l'analyse technique, qui doit réaliser des tests plus approfondis pour chaque solution, permet de découvrir ces logiciels inadaptés et de les éliminer de la liste des solutions candidates.

4.5.3 Quand opter pour cette méthode ?

A chaque fois qu'une migration d'un logiciel est envisagée. Comme cette étape ne demande pas beaucoup de temps, il est toujours intéressant de savoir ce que le marché offre comme possibilités avant de se lancer trop rapidement vers une solution qui pourrait ne pas convenir à moyen terme.

5 Analyse technique et fonctionnelle

5.1 Définition et objectifs

Cette étape consiste à tester en profondeur les différentes solutions candidates sur le matériel existant et avec les logiciels qui ne seront pas changés.

Les objectifs de cette méthode sont :

- Trouver les logiciels envisagés qui pourraient causer des problèmes d'ordre technique.

  Exemple : Si un logiciel maison fonctionne en étroite collaboration avec Microsoft Office, il se peut que passer à OpenOffice entraîne des problèmes techniques compliqués et financièrement non rentables.

- Trouver les coûts matériels que vont engendrer les différentes solutions candidates.

  Exemple : si on envisage de changer de suite bureautique, il se peut que les ordinateurs présents dans le système nécessitent une mise à jour de la mémoire vive ou le changement complet de certaines configurations matérielles qui ne supporteraient pas le logiciel envisagé car trop gourmand en puissance.

- Trouver les logiciels qui, après un test plus approfondi, ne correspondent pas aux besoins des utilisateurs.

  Exemple : il est possible qu'un logiciel soit sélectionné lors de la proposition de solutions candidates car il correspond aux besoins principaux des utilisateurs mais qui, au final, après certains tests plus poussés ne correspond en fait pas du tout aux attentes des utilisateurs.

Les deux premiers point correspondent à ce que nous pourrions nommer l'analyse technique et le troisième point à ce que nous pourrions appeler l'analyse fonctionnelle.

L'objectif attendu de cette étape est donc une estimation sur le coût en matériel de chaque solution. Et la cotation de chaque logiciel en fonction des besoins correctement satisfaits. Après cette étape, il est possible que certaines solutions candidates soient éliminées car elles ne remplissent pas suffisamment les besoins des utilisateurs.

5.2 Contexte d'utilisation de la méthode

Lors d'une migration, il est toujours nécessaire de réaliser des tests basés sur le matériel existant. En effet, passer de Windows à Linux peut entraîner des incompatibilités matérielles qu'il vaut mieux prévoir avant de généraliser la migration. Cela évitera des désagréments et des coûts cachés. La migration d'un seul logiciel peut aussi nécessiter des mises à jour matériel, peut entraîner des problèmes logiciels et peut nécessiter des formations. Ce qui est toujours intéressant de savoir avant de choisir une solution définitive.

Il est tout aussi intéressant de tester en profondeur les fonctionnalités des logiciels envisagés car un logiciel ne répondant pas aux besoins des utilisateurs peut entraîner le refus de la migration par ceux-ci et surtout entraîner une diminution de leur productivité.

5.3 Les acteurs

5.3.1 Qui participe à l'analyse technique ?

Les principaux participants à l'analyse technique sont les informaticiens de l'organisation qui envisage une migration.

Mais il est aussi nécessaire de faire appel aux utilisateurs pour qu'ils donnent leur point de vue sur les solutions envisagées. C'est grâce à leur avis que l'on saura si un nouveau logiciel rencontre leurs besoins fonctionnels.

Comment recruter et convier les participants ?

Le recrutement des informaticiens ne doit causer aucun problème car les tests de nouvelles solutions doivent faire partie de leurs compétences.

Le recrutement des utilisateurs peut par contre causer quelques difficultés mais dans l'ensemble, les utilisateurs experts et motivés voudront savoir ce qui leur sera proposé. L'appel à candidature est la meilleure solution pour trouver les plus motivés et donc les mieux placés pour tester les nouvelles solutions. Il peut aussi être intéressant de faire appel à des utilisateurs experts. En effet, pour tester un nouveau logiciel de création de sites Internet, il est intéressant de faire appel à des webmasters.

5.4 Mise en oeuvre de la méthode

5.4.1 Faut-il faire appel à un sous-traitant pour réaliser une analyse technique et fonctionnelle ?

Lors de cette étape, faire appel à un sous-traitant ne sera probablement pas un luxe. En effet, il faut admettre que nous n'avons pas toutes les connaissances et qu'il existe donc des personnes plus compétentes que nous dans certains domaines.

D'autant plus que c'est au contact de personnes plus compétentes que nous apprenons le plus. Donc avec l'aide d'un sous-traitant, vous aurez une analyse technique et fonctionnelle bien faite et vos employés ICT auront acquis de nouvelles connaissances.

5.4.2 Quelles sont les ressources nécessaires ?

La principale ressource sera le temps des informaticiens et des utilisateurs qui participeront à cette étape. Les ressources matérielles nécessaires ne devraient pas à priori être énormes.

L'appel à un sous-traitant coûte aussi de l'argent, et parfois beaucoup.

5.4.3 Combien de participants faut-il prévoir ?

Le nombre de participants surtout du côté des utilisateurs dépendra fortement du nombre d'applications qui devront être changées. Si nous n'envisageons que de modifier la suite bureautique, le nombre de participants peut-être limité à quelques personnes bien choisies. Tandis que si nous envisageons de modifier le système informatique dans son entièreté, le nombre de personnes ayant des compétences différentes et devant être impliquées sera beaucoup plus important.

5.4.4 Quelle est la durée de l'analyse technique et fonctionnelle ?

Cette étape peut durer un certain temps. Mais tout dépendra à nouveau de l'étendue de la migration envisagée.

5.4.5 Quel est le résultat de l'analyse technique et fonctionnelle ?

Les résultats attendus sont :

- un rapport sur les modifications matérielles qui devront être apportées dans le cas de la mise en production de chaque solution,
- une classification chiffrée des logiciels, par catégorie, envisagés dans les différentes solutions proposées, basées sur le respect des besoins fonctionnels des utilisateurs.

5.5 Atouts et limites

5.5.1 Atouts

La classification des logiciels et les coûts matériels envisagés pour la mise en place de chaque solution devraient être d'une aide précieuse pour les étapes suivantes que sont l'analyse économique et bien évidemment le choix d'une solution définitive.

5.5.2 Limites

Cette étape est limitée par le fait que nous ne testons les différentes solutions que dans un environnement bien défini. Pour connaître le fonctionnement réel de chaque solution, il faudrait réaliser des tests en environnement réel. Ce genre de test sera effectué sur la solution qui sera choisie. En effet, réaliser ces tests en environnement réel coûterait beaucoup trop d'argent et de temps à l'organisation.

5.5.3 Quand opter pour cette méthode ?

Cette étape est pour nous indispensable pour nous aider à choisir la solution finale qui sera implantée dans l'organisation.

6 Analyse économique

6.1 Définition et objectifs

Cette étape consistera à définir le TCO[56] et le ROI[57], Return On Investment, de chaque solution. Ces deux éléments permettront d'évaluer le « coût » de chaque solution envisageable.

6.2 Contexte d'utilisation de la méthode

Le TCO reprendra principalement les coûts suivants :

- le coût d'acquisition des licences,
- le coût d'acquisition de matériel (ce coût est calculable grâce aux premiers tests réalisés et donc à l'estimation des modifications matérielles à apporter pour installer la solution),
- le coût d'installation des logiciels (coût lié aux tests à réaliser sur la solution avant de l'installer et aux coûts de l'installation proprement dite),
- les coûts de formations (coûts liés à la formation de l'équipe IT et à la formation des utilisateurs),
- le coût lié à la reprise de la documentation.

Pour chaque solution possible, on calculera aussi son ROI en la comparant avec l'ancienne solution. Par la suite on pourra comparer le ROI de chaque solution entre elles. Et faire des simulations sur le long terme où certaines solutions sont beaucoup plus avantageuses d'un point de vue économique.

6.3 Les acteurs

6.3.1 Qui participe à l'analyse économique ?

L'équipe chargée de cette étape devrait être composées de personnes ayant des compétences diverses mais principalement économiques.

Nous conseillons donc de placer dans cette équipe :

- des experts informatiques connaissant parfaitement les différentes solutions envisagées,
- des analystes financiers spécialisés dans les calculs et simulations spécifiques à cette étape,
- des utilisateurs experts qui peuvent estimer la productivité des utilisateurs dans chacune des solutions à analyser.

Comme annoncé, nous avons donc une équipe pluridisciplinaire qui regroupe les différentes facettes des estimations à réaliser.

56 Voir Text 21 pour sa définition.
57 Voir Text 21 pour sa définition.

Le ROI (Return On Investment) selon [Guliani & Woods, 2005]

Le retour sur investissement est un calcul d'économiste important lors de tout investissement. Comme une migration est un investissement, elle n'échappe pas au calcul du ROI. Dans ce cas, ce n'est pas uniquement un calcul numérique. Il est autant basé sur des questions comme : « L'investissement à long terme est-il rentable en comparaison aux coûts d'entrée? », « La migration a-t-elle une utilité ? », « Quelle différence de coûts va-t-elle engendrer par rapport à l'ancien système? ».

Le ROI d'un système informatique est élevé si ce système apporte des avantages tant dans le fonctionnement de la société que du point de vue financier. En effet, pour calculer le ROI, on regarde si la nouvelle solution apporte des nouveautés utiles à l'utilisateur final. On regarde si l'ergonomie est suffisante par rapport à l'ancien système, si les fonctionnalités proposées sont « meilleures ». Et si elle ne coûte pas plus cher à l'emploi. Il faut donc que le nouveau système réponde aux besoins des utilisateurs et leur permette d'atteindre des conditions optimales de productivité sans coûter plus cher à la société.

C'est un peu comme dans le milieu de la construction où l'achat d'une nouvelle pelleteuse se joue au niveau financier mais aussi aux niveaux de la sécurité, de la productivité, etc. En effet, les questions posées sont : « Est-ce que la nouvelle pelleteuse, dans des conditions optimales, permet d'aller plus vite que l'ancienne? », « Est-ce que les ouvriers seront plus en sécurité qu'avant ? ».

On voit donc que le ROI est important tant au niveau financier qu'au niveau de la productivité. Dans le cas d'une migration, c'est la productivité et les coûts à long terme qui doivent être analysés.

Au niveau financier, pour calculer le ROI, il est intéressant de comparer le TCO (Total Cost of Ownership) de la solution actuelle et le TCO de la solution à mettre en place. En effet, si la nouvelle solution a un TCO plus faible que l'ancienne, cela ne peut apporter qu'un plus au ROI. En effet, si, à performance égale, l'utilisation d'une nouvelle solution coûte moins cher qu'une ancienne, la société gagne, à long terme, à utiliser cette nouvelle solution.

Le TCO (Total Cost of Ownership) selon [AWT, 2005] et [Aboubekr & Rivard, 2005]

Le TCO est le coût de possession d'un ordinateur. Il est composé de coûts du matériel, de coûts liés aux logiciels (principalement les licences), de coûts indirects (gestion administrative, installation, maintenance) et de coûts de support (aide, formation, etc.).

Le TCO doit être géré comme un plan comptable avec des postes budgétaires différents.

Il faut bien faire la différence entre le TOC et les coûts liés à la migration. Par exemple, si dans le TCO, il est prévu de modifier la RAM d'un ordinateur, il ne faut pas le comptabiliser dans les coûts de la migration pour autant que modifier la RAM soit nécessaire pour la migration.

Text 21: ROI and TCO

6.3.2 Comment recruter et convier les participants ?

Les membres de l'équipe chargée de l'analyse économique de chaque solution étant soit des membres du personnel soit des contractants, les amener à participer à cette activité ne devrait causer aucun problème.

6.4 Mise en oeuvre de la méthode

6.4.1 Faut-il faire appel à un sous-traitant pour réaliser une l'analyse économique ?

Si nous envisageons de ne changer, par exemple, que la suite bureautique, l'aide d'analystes externes sera probablement peu utile. Par contre si la migration envisagée est conséquente et implique de grandes sommes d'argent, faire appel à des experts sera sûrement une bonne solution si l'organisation ne possède pas les compétences requises en interne.

Il est évident qu'admettre que nous n'avons pas les compétences requises est souvent difficile mais permet de nous entourer de personnes compétentes qui nous aideront à prendre les bonnes décisions.

6.4.2 Quelles sont les ressources nécessaires ?

La principale ressource dont aura besoin l'équipe d'analyse sera l'information récoltée sur chaque solution. Elle devra aussi avoir accès aux ressources humaines nécessaires pour mener d'autres tests ou recherches sur les solutions envisagées si les informations disponibles ne sont pas suffisantes.

6.4.3 Comment préparer une analyse économique ?

Nous devons donner à l'équipe chargée de cette analyse tous les moyens et le soutien dont elle aura besoin pour mener à terme sa mission.

6.4.4 Combien de participants faut-il prévoir ?

Il est conseillé d'avoir une équipe assez réduite mais experte pour ce genre d'analyse.

6.4.5 Quelle est la durée de l'analyse économique ?

Cette étape étant importante pour le choix de la future solution, le temps imparti doit être suffisamment long pour permettre à ses protagonistes de réunir toutes les informations nécessaires et les comptabiliser.

Il vaut mieux consacrer un peu plus de temps que prévu pour cette étape que de regretter plus tard d'avoir choisi une mauvaise solution sur des estimations erronées car réalisées dans la précipitation.

6.4.6 Quel est le résultat de l'analyse économique ?

Le résultat attendu est une estimation, la plus précise possible et basée sur le TCO et le ROI, du « coût » de chaque solution envisagée.

6.5 Atouts et limites

6.5.1 Atouts

Cette étape doit nous aider à choisir une solution qui a un coût estimé faible par rapport aux avantages qu'elle apportera à l'organisation.

Cette étape est basée sur des principes, TCO et ROI principalement, qui ont déjà fait leurs preuves dans de nombreuses situations. Ce ne sont pas des concepts inconnus aux économistes.

6.5.2 Limites

La principale limite est que cette étape ne fournit qu'une estimation des coûts liés aux différentes solutions envisageables. Nous ne sommes donc pas à l'abri d'erreurs liées à une mauvaise appréciation des coûts liés à l'une ou l'autre solution.

6.5.3 Quand opter pour cette méthode ?

Cette étape est primordiale pour éviter de choisir une solution qui à priori coûterait plus chère pour moins d'avantages que d'autres.

Elle fait donc partie des prémices à l'étape du choix définitif de la nouvelle solution à implanter dans l'organisation.

6.6 Informations complémentaires

- [Guliani & Woods, 2005] et plus précisément le chapitre 4,
- [Aboubekr & Rivard, 2005]

7 Choix de la solution finale

7.1 Définition et objectifs

Maintenant que nous avons grâce aux étapes précédentes fait le tour des différentes solutions candidates, et que certaines peuvent être éliminées immédiatement car :

- elles sont beaucoup trop chères par rapport aux avantages qu'elles apportent,
- elles n'offrent aucun avantage par rapport à l'ancienne solution.

Il est temps de choisir la solution qui sera mise en place.

La solution qui sera choisie est celle qui apporte le maximum d'avantages en coûtant le moins cher. C'est une solution qui sera évolutive par rapport à l'évolution des technologies.

Il va aussi falloir justifier le choix de cette solution par rapport aux autres. Cette justification permettra de mettre le plus de personnes en accord avec ce choix.

7.2 Contexte d’utilisation de la méthode

Une fois les étapes préliminaire au choix de la nouvelle solution ont été réalisées, il est temps de prendre une décision, soit continuer la migration et donc choisir une des solutions soit arrêter la migration car les solutions proposées ne sont pas valables. Il faudra alors tout reprendre à zéro (dans ce cas, on peut facilement passer outre les étapes d'analyse de l'existant).

Chaque participant à cette étape devrait être amené à proposer les différentes solutions qu'il trouve intéressantes et pourquoi. Ensuite, tout le monde serait amené à voter pour choisir la solution qu'ils trouvent la plus avantageuse pour l'organisation.

7.3 Les acteurs

7.3.1 Qui participe au choix de la solution finale ?

Les personnes les mieux placées sont les responsables de la migration ainsi que la direction qui devra donner son accord aux différents investissements qu'il faudra réaliser pour mettre en place la nouvelle solution.

7.3.2 Comment recruter et convier les participants ?

Les participants à cette étape devraient avoir été choisis dès la décision du lancement de la recherche d'une nouvelle solution. Ils devraient aussi avoir suivi l'évolution de cette recherche. En général une réunion ou une série de réunions devraient suffire pour choisir la nouvelle solution.

7.4 Mise en oeuvre de la méthode

7.4.1 Quelle est la durée du choix de la solution finale ?

La durée de cette étape dépendra de plusieurs facteurs :

- la qualité des différentes solutions mises en concurrence,
- le nombre de participants à ce choix,
- la mise en accord sur un choix qui fait l'unanimité.

Ces trois facteurs nous paraissent les plus importants pour calculer la durée de cette étape. En effet, s'il n'y a qu'une solution qui parait intéressante à la majorité des protagonistes de cette étape, le choix sera vite fait. Si par contre il existe plusieurs solutions intéressantes et qu'il y a beaucoup de personnes autour de la table, trouver un choix commun sera probablement plus difficile et demandera probablement plus de temps.

7.4.2 Quel est le résultat du choix de la solution finale ?

Le résultat attendu est donc le choix d'une solution et l'accord du comité de migration pour la poursuivre. Cette étape a aussi pour but de dégager les moyens nécessaires à la poursuite de la migration.

Il est aussi possible qu'aucune solution ne soit satisfaisante alors les équipes de migrations retravailleront les propositions et soumettront ultérieurement d'autres solutions.

Il est aussi possible que plusieurs solutions soient retenues pour des tests et des analyses complémentaires afin de pouvoir les départager selon d'autres critères.

7.5 Atouts et limites

Nous ne pouvons pas déterminer d'atouts ou de limites à cette étape car elle est obligatoire dans tout processus de migration. En effet, il y a toujours un moment où la direction de l'organisation et le comité de migration doivent se réunir pour décider de la suite à donner aux études préliminaires. Et cette étape est le meilleur moment pour ce faire.

Cette l'étape peut être définies comme l'étape « Go or No Go » d'une migration.

8 Présentation de la nouvelle solution

8.1 Définition et objectifs

Cette étape a pour but de faire connaître la nouvelle solution aux futurs utilisateurs et leur permettre d'adhérer aux motifs ayant poussé le comité de migration à choisir cette solution plutôt qu'une autre.

8.2 Contexte d'utilisation de la méthode

Lors de toute migration, il est primordial que les futurs utilisateurs d'une solution se sentent écoutés et impliqués dans son évolution. Il est aussi nécessaire qu'ils adhèrent à la nouvelle solution pour que son implantation ne soit pas un échec.

Les utilisateurs sont en première ligne et donc il faut les motiver à accepter les futurs changements.

8.3 Les acteurs

8.3.1 Qui participe à la présentation sur la nouvelle solution ?

Les présentateurs de la nouvelle solution doivent être des personnes influentes auprès des utilisateurs. Nous pouvons soit nous baser sur des informaticiens reconnus pour leur compétence au sein de l'organisation soit sur des utilisateurs experts qui ont participé aux tests de la nouvelle solution soit encore à des personnes extérieures qui ont une expérience de cette solution.

Les personnes à qui est destinée cette présentation sont les futurs utilisateurs de la nouvelle solution. Les différents destinataires concernés par la présentation seront par exemple :

soit tous les employés de l'organisation si une nouvelle suite bureautique est envisagée,

soit les responsables de la maintenance les développeurs web si la mise en place d'un nouveau serveur Internet ou Intranet est choisie.

8.3.2 Comment recruter et convier les participants ?

Généralement le personnel d'une organisation apprécie qu'on le tienne au courant des changements qui les concernent. C'est pourquoi les convier à participer à la présentation ne devrait pas poser de grandes difficultés.

Quant aux présentateurs, ils doivent être au courant dès le lancement de la procédure de migration qu'ils seront invités à présenter la solution finalement choisie. Et leur participation ne posera probablement pas de problèmes.

8.4 Mise en oeuvre de la méthode

8.4.1 Faut-il faire appel à un sous-traitant pour réaliser la présentation de la nouvelle solution ?

Dans certains cas, la présentation par un sous-traitant expert pour tout ou une partie de la nouvelle solution sera nécessaire pour la présenter correctement. Il ne faut pas avoir peur de dépenser un peu d'argent car une présentation par une personne connaissant son sujet à fond porte toujours ses fruits. En effet si la présentation est un échec, la mise en place de la nouvelle solution risque de se faire dans la douleur.

8.4.2 Quelles sont les ressources nécessaires ?

En fonction du type de migration visée, une nouvelle application utilisée par un grand nombre ou par quelques experts ou bien le remplacement d'une grande partie du système existant, les ressources envisagées pourront fortement varier. En effet, la présentation d'un nouveau logiciel à cinq personnes demande moins de ressources que la présentation d'une application à l'ensemble du personnel.

8.4.3 Combien de participants faut-il prévoir ?

Le nombre de participants sera fortement dépendant comme pour le point précédent du type de migration qui est visée.

8.4.4 Quelle est la durée de la présentation de la nouvelle solution ?

En fonction du type de migration visée, nous pouvons présenter la nouvelle solution soit à tous ses futurs utilisateurs en une fois. Dans ce cas, le temps imparti comprendra le temps préparatoire et le temps de la présentation. Soit à des petits groupes pour que l'interaction soit plus importante et dans ce cas, la durée dépendra en plus du nombre de présentations à réaliser.

8.4.5 Quel est le résultat de la présentation de la nouvelle solution ?

Le résultat attendu par cette présentation est l'acceptation par les futurs utilisateurs de la nouvelle solution.

8.5 Atouts et limites

8.5.1 Atouts

Cette présentation a comme atout principal de faire connaître aux futurs utilisateurs de la nouvelle solution leur futur environnement de travail. Et ainsi de connaître leurs sentiments à l'égard de cette solution. Nous pourrons alors les rassurer et les amener à supporter cette migration.

8.5.2 Limites

Si la présentation se passe mal ou si la nouvelle solution est mal présentée, les utilisateurs vont craindre le changement et vont être particulièrement réticents à la mise en place de cette nouvelle solution. Et la migration risque d'être un échec tant technique que financier.

Rattraper une première mauvaise impression n'est jamais évident. Il est donc conseillé de tout mettre en place pour ne pas rater cette première présentation.

8.5.3 Quand opter pour cette méthode ?

Cette étape est d'autant plus importante qu'elle touche un grand nombre de personnes de l'organisation. Si le nombre de personnes touchées par la migration est minime, il est fort possibles que ces personnes soient déjà intervenues lors des étapes précédentes et qu'elles connaissent déjà le but de cette migration. Ce qui ne sera évidemment pas le cas pour une migration touchant pratiquement l'ensemble du personnel.

9 Formation de l'équipe IT

9.1 Définition et objectifs

Cette étape est préalable à la phase de tests prévue avant la mise en production de la nouvelle solution. Si la nouvelle solution est dans la lignée de celle qui va être modifiée, il est possible que cette étape ne soit pas nécessaire. Par exemple, une migration de logiciel utilisateur (Microsoft Office vers OpenOffice ne demandera pas une formation complémentaire de l'équipe IT pour qu'elle puisse gérer son implantation. Par contre une formation au nouvel outil sera probablement utile. Mais cela rentre plus dans l'étape de formation des utilisateurs.

Par contre si la nouvelle solution est basée sur une technologie différente de l'ancienne solution, il est plus que probable que l'équipe IT de l'organisation ait besoin d'une formation plus ou moins poussée pour être apte à travailler correctement dans son nouvel environnement. Par exemple, le passage de Windows à Linux demandera une formation de l'équipe IT pour qu'elle soit compétente pour mener les tests et l'implantation de ce nouvel environnement.

9.2 Les acteurs

Les acteurs de cette formation sont bien évidement les membres de l'équipe IT et plus particulièrement les personnes qui devront travailler sous le nouvel environnement.

Les autres acteurs de cette étape sont les formateurs. Il existe en Belgique de nombreuses sociétés qui sont spécialisées dans la formation.

9.3 Mise en oeuvre de la méthode

9.3.1 Faut-il faire appel à un sous-traitant pour réaliser un la formation de l'équipe IT ?

Il est conseillé de faire appel à un formateur externe et ce principalement si l'organisation migre vers une technologie qui ne fait pas partie de ses compétences de base.

9.3.2 Combien de participants faut-il prévoir ?

Le nombre de participants dépendra du nombre de représentants de l'équipe IT qu'il sera nécessaire de former.

9.3.3 Quelle est la durée de la formation de l'équipe IT ?

La durée de la formation dépendra fortement de la complexité de la technologie employée et de sa différence avec celle employée jusqu'à présent dans l'organisation.

9.3.4 Quel est le résultat de la formation de l'équipe IT ?

Le résultat attendu est une équipe IT performante sous le nouvel environnement.

9.4 Atouts et limites

9.4.1 Atouts

La formation permettra à l'équipe IT d'éviter de travailler avec la méthode « essai erreur » et d'être après la formation rapidement plus productive que si elle devait apprendre à utiliser la nouvelle technologie par elle-même.

Cette formation aura aussi le mérite de mettre tout le monde sur le même pied et éviter à certaines personnes de se croire plus compétentes et par conséquence commettre des erreurs importantes.

Sans formation, le risque est grand que l'équipe IT se sente plus compétente que ce qu'elle est effectivement et nous tombons alors dans les risques liés à la compétence des équipes IT présentées dans le Text 8 au troisième chapitre, page 50.

9.4.2 Limites

Malheureusement, cette formation n'élimine pas entièrement le risque cité précédemment. Mais elle permet de le limiter.

La formation se concentre généralement sur une partie de la technologie et sur des exercices « simples ». Il est donc fort probable que l'étape suivante soit le lieu de la mise en pratique de cette formation et que l'équipe IT doive encore apprendre beaucoup de choses par elle-même.

9.4.3 Quand opter pour cette méthode ?

Il est généralement important de bien estimer les compétence de son équipe IT. Cela nous permettra de lui proposer des formations adéquates et lui éviter de perdre son temps avec des formations inutiles. Nous lui proposeront donc une formation si nous estimons qu'elle en a besoin.

Généralement, tous les membres d'une équipe IT n'ont pas les mêmes compétences, compétences qui dépendent de leur parcours professionnel. Il est donc nécessaire de proposer à chacun la formation qui lui conviendra le mieux.

Cette méthode est principalement à conseiller lors d'un changement important de technologie (passage de Windows Linux par exemple).

10 Tests d'intégration de la nouvelle solution

10.1 Définition et objectifs

L'objectif de cette étape est de tester la nouvelle solution dans ses moindres détails. Ainsi, on pourra trouver et corriger les problèmes d'intégration et/ou techniques qui pourraient survenir. Ces tests doivent suivre un plan précis afin d'éviter à l'équipe d'oublier certains tests importants.

L'objectif est donc de limiter les imprévus lors de la mise en place de la solution dans l'organisation.

10.2 Contexte d'utilisation de la méthode

Nous avons déjà, lors des étapes précédentes, testé les différentes solutions qui étaient en concurrence. Maintenant qu'une de ces solution a été choisie, il est nécessaire et indispensable de la tester beaucoup plus en détail.

De plus il est conseillé de tester la solution dans un environnement proche de l'environnement réel. C'est-à-dire où tous les ingrédients du système d'information de l'organisation sont représentés. Par exemple, si l'organisation possède des serveurs de données qui ne seront pas changés, il faut alors vérifier que le nouveau système communique bien avec eux et, si ce n'est pas le cas, trouver une solution à ce problème.

Pour nous aider à réaliser cette étape, il existe de nombreux documents et guides de bonne pratique disponibles.

10.3 Les acteurs

10.3.1 Qui participe aux tests d'intégration de la nouvelle solution ?

Les personnes les mieux placées pour tester la nouvelle solution et trouver les problèmes que son implantation pourrait causer sont les membres de l'équipe IT et les utilisateurs intégrés à l'équipe dirigeant la migration s'il y en a une.

10.3.2 Comment recruter et convier les participants ?

Il n'a a rien à mettre en place pour recruter les participants à cette étape. Ils s'y sont engagés en rejoignant l'organisation ou l'équipe responsable de la migration.

10.4 Mise en oeuvre de la méthode

10.4.1 Faut-il faire appel à un sous-traitant pour réaliser les tests d'intégration de la nouvelle solution ?

Par contre l'aide d'un sous-traitant ou d'un prestataire externe sera plus que nécessaire et ce principalement dans le cas où l'équipe IT n'est pas suffisamment compétente pour mettre en place seule la nouvelle solution.

Faire appel à un sous-traitant quand on n'est pas certain d'avoir les compétences est une preuve d'intelligence ou devrait être vu comme tel.

10.4.2 Quelles sont les ressources nécessaires ?

Les ressources nécessaires pour réaliser cette étape dépendront à nouveau de l'importance de la migration à réaliser. Les ressources humaines utiles seront dans la continuité de ce qui a été utile précédemment. Par contre il est a prévoir une augmentation des ressources matérielles nécessaire et ce principalement pour les tests en environnement « réel ».

10.4.3 Comment préparer les tests d'intégration de la solution finale ?

L'équipe IT en ayant suivi une formation aux technologies de la nouvelle solution devrait être armée pour faire face au défi de tester la nouvelle solution. Elle devra toutefois continuer à se former par ses propres moyens tout au long de la période de tests.

L'équipe de test aura besoin d'un plan de test précis. Cela lui permettra de savoir que faire et quand le faire.

10.4.4 Quelle est la durée des tests d'intégration de la nouvelle solution ?

La durée de ces tests dépendra évidemment de la complexité de la solution à mettre en place. Et elle dépendra aussi du temps pour résoudre les problèmes rencontrés par l'équipe de test.

10.4.5 Quel est le résultat des tests d'intégration de la nouvelle solution ?

Les résultats attendus de ces tests sont :

- le nouveau système prêt à être implanté dans l'organisation,
- une solution pour tous les risques envisagés,
- et la mise sur pied de scénarios de sauvegarde au cas où un problème non envisagé survienne.

10.5 Atouts et limites

10.5.1 Atouts

Grâce à ces tests la majorité des risques devrait avoir été découverte et une solution doit être prête. On a donc éliminé toutes les surprises ou presque. On a même envisagé un plan de secours dans le cas ou un risque non découvert survenait.

Si durant cette phase on découvre un problème techniquement non soluble ou trop cher à solutionner, on peut toujours faire marche arrière. Cette marche arrière, en période de tests, étant beaucoup moins grave que si on avait déjà entamé l'implantation de la nouvelle solution.

10.5.2 Limites

Comme dit dans les atouts, on n'est pas à l'abri d'un problème non envisagé qui pourrait mettre en péril l'implantation de la solution.

Les tests en environnement « réel » restent en effet des tests et nous devrons donc toujours rester prudents et attentifs à tous les événement qui pourraient survenir lors de la mise en place de la nouvelle solution pour être prêts à intervenir.

10.5.3 Quand opter pour cette méthode ?

Lors de toute migration il est recommandé de tester la nouvelle solution dans un environnement « réel » simulé. Cette période de test permet généralement de découvrir quelques problèmes qui auraient pu faire échouer la mise en place du système s'ils n'avaient pas été découverts avant.

11 Formation des utilisateurs

11.1 Définition et objectifs

Les principaux objectifs de la formation des utilisateurs sont :

- les former aux nouveaux outils,
- les soutenir dans le changement pour qu'ils ne soient pas perdus dans le nouvelle environnement,
- leur fournir les moyens nécessaires à conserver leur niveau de productivité.

La formation permettra surtout aux utilisateurs de ne pas être perdus dans leur nouvel environnement de travail.

11.2 Contexte d’utilisation de la méthode

Cette étape doit intervenir avant la mise en production ou dès son début. Si ce n'est pas le cas, les utilisateurs ne vont pas apprécier la nouvelle solution et vont la rejeter. En plus s'ils ne sont pas formés, leur productivité risque de chuter car ils ne sauront plus comment travailler.

Il est évident que chaque migration n'appelle pas forcément une formation. Tout dépend évidement de ce qui va être modifié. Nous devrons donc analyser la situation au cas par cas. En nous basant sur les tests réalisés par les utilisateurs volontaires lors de l'analyse

fonctionnelle de la solution.

11.3 Les acteurs

11.3.1 Qui participe à la formation des utilisateurs ?

Les utilisateurs de la nouvelle solution sont bien évidement les principaux acteurs de cette étape. Mais sans formateurs, ils auront difficile à être formés.

Les acteurs de cette étape sont donc les utilisateurs et les formateurs.

11.3.2 Comment recruter et convier les participants ?

Les utilisateurs seront généralement obligés de participer à la formation donc les recruter ne devrait poser aucun problème.

Le recrutement des formateurs ne devrait pas poser non plus beaucoup de problèmes. En effet, si nous faisons appel à des formateurs professionnels, leur recrutement passe par la signature d'un contrat. Si la formation est donnée par un membre du personnel, cette personne est probablement un volontaire et donc son recrutement est lié à la motivation des personnes aptes à donner la formation.

11.4 Mise en oeuvre de la méthode

11.4.1 Faut-il faire appel à un sous-traitant pour réaliser la formation des utilisateurs ?

Faire appel à un sous-traitant sera souvent la meilleure solution pour former les utilisateurs à des applications complexes. Par contre, pour de petites applications ou des applications fortement intuitives, une formation par un membre de l'équipe IT sera probablement suffisante.

Par exemple, pour former les utilisateurs à OpenOffice, faire appel à un formateur habitué à ce logiciel est sûrement la meilleure solution. Par contre, faire appel à un formateur professionnel pour former les utilisateurs à Firefox est probablement inutile.

11.4.2 Quelles sont les ressources nécessaires ?

Ces formations coûtent généralement du temps et de l'argent à une organisation. L'argent correspond à la non productivité du personnel qui participe à la formation et au prix de la formation en elle-même.

La formation coûte aussi du temps de travail aux participants à cette formation.

11.4.3 Comment préparer la formation des utilisateurs ?

Il est important que le personnel sente cette formation comme un plus pour leurs compétences et leur évolution dans l'organisation. La préparation de la formation consistera donc principalement à informer les utilisateurs des raisons de la formation et des avantages qu'ils tireront à la suivre.

11.4.4 Combien de participants faut-il prévoir ?

Normalement, la formation doit être donnée à toutes les personnes susceptibles d'utiliser la nouvelle solution. Si dans une solution, il y a différentes applications, tous ne devront probablement pas être formés à toutes les applications mais à celles qui feront partie de leur environnement de travail quotidien.

11.4.5 Quelle est la durée de la formation des utilisateur ?

Il est impossible de répondre de manière précise à cette question. En effet, la durée de la formation dépendra du nombre d'utilisateurs à former, on ne peut pas former 500 personnes en même temps... Et du niveau de la formation, le temps de formation pour Firefox est par exemple beaucoup moindre que pour OpenOffice.

11.4.6 Quel est le résultat de la formation des utilisateurs ?

Les utilisateurs sont formés et prêts à utiliser les outils proposés dans la nouvelle solution.

11.5 Atouts et limites

11.5.1 Atouts

La formation des utilisateurs quand elle est nécessaire leur permet d'être directement à l'aise avec la nouvelle solution et la diminution de productivité liée à leur adaptation à leur nouvel environnement sera beaucoup plus faible que s'ils n'avaient pas eu de formation.

La formation permet aussi aux utilisateurs de se sentir concernés par la migration. En sachant qu'on est attentif à leurs préoccupations, ils accepteront, généralement, beaucoup plus facilement les petits problèmes et désagréments liés à la jeunesse de la nouvelle solution.

11.5.2 Limites

Ce n'est malheureusement pas une formation qui va nous permettre de faire accepter la nouvelle solution aux plus réfractaires des utilisateurs.

Il est aussi évident que la productivité des utilisateurs va légèrement baisser lors de leur prise en main de la nouvelle solution.

Ces deux limites sont momentanées. La productivité des utilisateurs devrait revenir à son niveau normal une fois qu'ils auront bien en main la solution et la mise en place d'une formation accélère souvent cette prise en main. Et pour ce qui est des réfractaires, chaque organisation doit faire avec et le temps aide généralement à atténuer les choses.

11.5.3 Quand opter pour cette méthode ?

Il faut opter pour cette méthode quand la nouvelle solution apporte des changements importants dans l'environnement de travail des utilisateurs. En effet, si la migration consiste à installer une nouvelle version de la suite bureautique utilisée depuis de nombreuses années, une formation ne sera peut être pas utile. Par contre si on change la suite bureautique, une formation sera probablement nécessaires pour que les utilisateurs ne perdent pas leur temps à trouver ce dont ils ont besoin.

Formation des utilisateurs

Et si la migration n'est pas visible des utilisateur (une migration du serveur mail par exemple), une formation des utilisateur est totalement inutile.

12 Mise en place de la nouvelle solution

12.1 Définition et objectifs

Cette étape est la dernière de la migration proprement dite. Elle consiste donc à mettre en production la nouvelle solution. Nous conseillons toutefois quelque temps après cette étape de réaliser un retour d'expérience auprès de l'équipe chargée de la migration et des utilisateurs pour voir ce qui a bien fonctionné et ce qui pourrait être amélioré lors des futures migrations que l'organisation sera amenée à réaliser.

12.2 Contexte d'utilisation de la méthode

Nous vous conseillons de réaliser la mise en production de la solution par différentes étapes. Nous allons ici présenter l'ordre que nous préconisons :

1. migrer le back-office, c'est-à-dire tout ce qui est invisible à l'utilisateur. Cela comprend notamment les serveurs tant de soutien que les serveurs applicatifs.
2. migrer le front-office et ce en commençant par les services les moins communicants et les plus intéressés par la migration. C'est-à-dire commencer par les services qui ne sont pas récalcitrant à la migration et qui pourront donc montrer l'exemple et donner envie aux autres services. Et par services moins communicant, nous voulons dire les services qui échangent peu d'informations numériques avec les autres services.
3. étende la migration du front-office aux services les plus communicants et/ou les plus récalcitrants à la migration. En effet, avec le temps, les services les plus récalcitrant verront que la migration se déroule bien et ils auront peur d'avoir été oubliés et nous demanderont de nous occuper d'eux.

12.3 Les acteurs

12.3.1 Qui participe à la mise en place de la nouvelle solution ?

Les principaux acteurs de cette étape sont les membres de l'équipe chargée de la migration et, bien évidemment, les utilisateurs de cette nouvelle solution.

12.3.2 Comment recruter et convier les participants ?

Cette étape fait partie du travail de l'équipe de migration et leur participation va donc de soi. Quant aux utilisateurs, ils n'auront pas le choix.

12.4 Mise en oeuvre de la méthode

12.4.1 Faut-il faire appel à un sous-traitant pour réaliser la mise en place de la nouvelle solution ?

Il sera parfois nécessaire de faire appel à des experts extérieurs et ce principalement si nous avons eu recours à leurs services dans les étapes précédentes. Plus les personnes travaillant à la migration seront compétentes plus elle se fera en douceur.

12.4.2 Quelles sont les ressources nécessaires ?

Les principales ressources nécessaires seront les ressources humaines mises à disposition pour réaliser cette étape ainsi que les ressources matérielles prévues.

La mise à jour matérielle prévue lors du choix de la solution doit être mise en place durant cette étape.

12.4.3 Comment préparer la mise en place de la nouvelle solution ?

Il est nécessaire de préparer cette migration en mettant au point un plan précis de ce qui doit être migré et dans quel ordre. Rien ne doit être laissé au hasard.

Plan de mise en production :

1. mise en place des serveurs 1 et 2,
2. mise en place des serveurs 4 et 7,
3. mise en place des serveurs 3, 5 et 6,
4. migration du service 8,
5. migration du service 1 et 5,
6. etc.

Text 22: exemple d'un plan de migration

12.4.4 Combien de participants faut-il prévoir ?

Le nombre de participants à cette mise en production dépendra du nombre de personnes touchées par cette migration. Le temps que chaque participant dépensera pour la migration sera lui dépendant de son rôle lors de la migration (membre actif du groupe chargé de la migration ou simple utilisateur la « subissant »).

12.4.5 Quelle est la durée de la mise en place de la migration ?

La migration sera plus ou moins longue en fonction de son importance. Mais il est conseillé de prévoir un temps suffisamment important pour ne pas presser l'équipe de migration. Car c'est toujours quand on est pressé que l'on commet le plus d'erreurs et l'accumulation de petites erreurs peut entraîner de graves problèmes.

12.4.6 Quel est le résultat de la mise en place de la nouvelle solution ?

De cette étape, nous attendons une nouvelle solution fonctionnelle au sein de l'organisation.

12.5 Quand opter pour cette méthode ?

Cette étape doit avoir lieu chaque fois qu'on décide de mettre en production une nouvelle solution.

12.6 Informations complémentaires

- [Oettinger, 1999-2001]

Conclusion

Dans cette conclusion, nous voudrions reprendre de manière structurée les trois termes qui ont guidé notre démarche.

Le premier est le « quoi ». Il s'agissait d'abord de clarifier la définition, les principes et les outils qui régissent l'open source. En effet, une partie de la frilosité des administrations à l'égard de l'open source provient d'un manque de connaissances. Dans cette partie, nous avons pu aborder la philosophie attachée à l'Open Source, les licences libres, les communautés Open Source, le mode de développement des logiciels Open Source et les différentes catégories de formats de fichiers. Nous avons aussi constaté que les termes « Open Source » et « Logiciel Libre » sont quasiment synonymes.

Le second est le « pourquoi ». Ce point avait comme objectif d'apporter aux administrations publiques une meilleure connaissance des enjeux que représente l'Open Source. A ce niveau, nous voudrions insister sur l'un des éléments mis en évidence, à savoir la non-dépendance des administrations publiques à l'égard des formats propriétaires. En effet, c'est selon nous, pour les administrations publiques, un enjeux majeur, dans la mesure où les données qu'elles traitent sont des biens publics qui, en tant que tel, ne peuvent être confiés à des formats propriétaires au risque de ne plus y avoir accès dans quelques années.

Enfin, le troisième est le « comment ». C'est sans doute la partie la plus innovante de ce mémoire dans la mesure où les méthodologies de migration sont relativement absentes dans la littérature consacrée à l'Open Source. Grâce à ce guide d'étapes, nous avons mis en évidence certains points dont :

- l'importance de savoir d'où l'on vient et où l'on va,
- l'importance et l'influence des facteurs humains dans une migration,
- et la nécessité de tester différentes solutions pour trouver celle qui conviendra le mieux.

Certes, certaines étapes du guide devraient faire l'objet d'un approfondissement, mais nous avons néanmoins réussi à dresser le cadre d'une démarche complète qui pourra, à l'avenir, structurer la migration des administrations publiques vers l'Open Source. En effet, une migration ne s'improvise pas et demande une démarche bien structurée.

Ce mémoire, vous l'aurez compris, n'est pas un mémoire technique mais bien plus un mémoire qui met en scène des compétences de gestion indispensables au métier d'informaticien. En effet, durant notre parcours universitaire, nous avons appris à développer et à déployer des logiciels. Nous avons aussi abordé la gestion de projets informatiques. Cependant, nous n'avons quasiment pas abordé les enjeux, les risques et les méthodologies liés aux migrations, que nous risquons, néanmoins, de rencontrer dans l'exercice de nos fonctions futures.

Bibliographie

1 Livres

- Auteur, *titre*, nom éditeur, lieu édition, date
- Eric S. Raymond, *The cathedral and the Bazaar*, O'Reilly, Sebastopol, 2001.
- Andrew M. St. Laurent, *Understanding Open Source and Free Software Licensing*, O'Reilly, Sebastopol, 2004.
- Gautam Guliani & Dan Woods, *Open Source for the Enterprise*, O'Reilly, Sebastopol, 2005.
- Stefan Koch, *Free/Open Source software Development*, Idea group publishing, London, 2005.
- Chris DiBona, Danese Cooper and Mark Stone, *Open Source 2.0*, O'Reilly, Sebastopol, 2005.
- Philippe Logerot, *Linux ou Windows Guide D'Aide à la Décision*, Dunod, Paris, 2003.
- B. Choppy, *Guide IDA de migration vers l'Open Source*, netproject Ltd, Morden, 2003.
- Malika Aboubekr & Suzanne Rivard, *Migration vers OpenOffice.org sous environnement Windows : analyse d'expériences internationales*, CIRANO, Montréal, 2005.
- CSC, Open Source: Open for Business, Computer Sciences Corporation, Aldershot, 2004.
- Terminal, *Logiciels libres : de l'utopie au marché*, L'Harmattan, Paris, 1999.
- Emmanuel Oettinger, *Considération de la migration d'un système opérationnel chez WWF International*, mémoire présenté à l'université de Lausanne, École des hautes études commerciales, Année académique 1999-2001.
- William Gonzalez, *Migration de la suite bureautique Microsoft Office vers OpenOffice.org*, document réalisé pour la Ville de Fleury-Les-Aubrais dans le cadre d'une formation en vue de l'obtention d'une titularisation, 2005.

2 Articles

- Etienne Montero, « Réflexions de synthèse – licences de logiciel libre : du neuf avec du vieux ? », extrait du livre Les logiciels libre face au droit, *Cahiers du CRID n°25*, 2005
- Josh Lerner and Jean Triole, « The Simple Economics of Open Source », *Journal of Industrial Economics n°52*, 197-234, June 2002.
- Sumanta Deb & Manish Kumar Sirvastava, « Managing Linux Migration », *Cutting Edge (Infosys)*, mars 2004.
- Maurice Svay - Jocelyn Aubert - Laurence Jacquet - Nicolas Gramolini, « La migration vers les logiciels libres est-ce raisonnable pour une entreprise? »,

MigrationLibre.pdf sur http://migrationlibre.free.fr/index.html, janvier 2005.

- Denis Bodor et Fleur Brosseau, « Firefox, plébiscité par la gendarmerie : interview », *Linux Pratique n°33*, pages 18-20, janvier-février 2006.

3 Articles sur Internet

Référence - Titre - Auteur	*Site Internet*	*Date de visite*
[AWT, 2005], Fiche de l'AWT : Le logiciel libre	http://www.awt.be/web/fic/index.aspx?page=fic,fr,m00,014,001 ou http://www.awt.be/contenu/tel/fic/m00,014.pdf	05/10/2005
[Viseur], Fiche 132 : La dynamique open source, Robert Viseur	http://www.logiciellibre.net/download/fiche132.pdf	05/10/2005
[Milan], La gendarmerie française fait le choix d'une bureautique libre, Anthony Milan	http://www.infogiciel.info/article0168.html	13/12/2005
The approved Licenses	http://www.opensource.org/licenses/index.php	29/09/2005
Why "Free" Software is too Ambiguous	http://www.opensource.org/advocacy/free-notfree.php	28/09/2005
The Open Source Definition, Bruce Perens	http://www.opensource.org/docs/definition.php	28/09/2005
Qu'est-ce qu'un logiciel libre, Karl Pradène ;	http://www.gnu.org/philosophy/free-sw.fr.html	28/09/2005
The Free Software Definition, Novalis	http://www.gnu.org/philosophy/free-sw.html	28/09/2005
Pourquoi « Free Software » est-il meilleur que « Open Source » , Benjamin Drieu	http://www.gnu.org/philosophy/free-software-for-freedom.fr.html,	28/09/2005
Why ``Free Software'' is better than ``Open Source'', Novalis	http://www.gnu.org/philosophy/free-software-for-freedom.html	28/09/2005
Les licences commentées	http://www.gnu.org/licences/licences-list.fr.html	29/09/2005
Various Licenses and Comments about Them	http://www.fsf.org/licensing/licenses/index_html	08/04/2006
Catégories de logiciels libres et non libres	http://www.gnu.org/philosophy/categories.fr.html	26/09/2005

Référence - Titre - Auteur	*Site Internet*	*Date de visite*
Terminology Wars: A Web Content Analysis, Eric S. Raymond	http://www.catb.org/~esr/writings/terminology	28/09/2005
Richard Stallman's Personal Home Page, Richard Stallman	http://www.stallman.org/	02/11/2005
Bruce Perens	http://perens.com/	02/11/2005
Tim O'Reilly Official Bio	http://www.oreilly.com/oreilly/tim_bio.html	02/11/2005
Les logiciels libres, François Guely	http://www.futura-sciences.com/comprendre/d/dossier50-1.php	03/11/2005
Firefox dépasse les 20% en Europe	http://www.xitimonitor.com/etudes/equipement13.asp	06/04/2006
Histoire de php	http://fr.php.net/history	06/04/2006
Introduction à Perl	http://www.commentcamarche.net/perl/perlintro.php3	06/04/2006
Le navigateur Firefox grignote… grignote toujours, Yves Grandmontagne	http://www.silicon.fr/articles/10880/Le-navigateur-Firefox-grignote%E2%80%A6-grignote-toujours.htm	06/04/2006
Market Share for Top Servers Across All Domains August 1995 - March 2006	http://news.netcraft.com/archives/2006/03/index.html	06/04/2006

4 Sites Internet Génériques

Nom du Site Internet	*Adresse Internet*
Projet GNU	http://www.gnu.org
Open Source Initiative	http://www.opensource.org
AWT	http://www.awt.be
Wikipedia Anglais	http://www.wikipedia.org
Wikipedia Français	http://fr.wikipedia.org
Projet Debian	http://www.debian.org
FSF	http://www.fsf.org
FSF France	http://www.fsffrance.org.
BSD	http://www.bsd.org
Ubuntu	http://www.ubuntu-fr.org et http://www.ubuntu.com

Sites Internet Génériques

Nom du Site Internet	*Adresse Internet*
Kubuntu	http://www.kubuntu.org
Red Hat	http://www.redhat.com/, http://www.fr.redhat.com/ et http://www.europe.redhat.com/belgium/
Suse	http://www.novell.com/linux/ et http://www.suse.com/fr/
IBM Linux Portal	http://www-1.ibm.com/linux/
Mozilla	http://www.mozilla.org
Mandriva (anciennement Mandrake)	http://www.mandriva.com/fr/community/ et http://www.mandriva.com/community/
OpenOffice.org	http://www.openoffice.org/ et http://fr.openoffice.org/
Lea web site	http://lea-linux.org
Perl	http://www.perl.org
Php	http://www.php.net
Xiti Monitor	http://www.xitimonitor.com
Licence GPL	http://www.opensource.org/licenses/gpl-license.php
Licence LGPL	http://www.opensource.org/licenses/lgpl-license.php
Licence MPL 1.1	http://www.opensource.org/licenses/mozilla1.1.php
Licence BSD	http://www.opensource.org/licenses/bsd-license.php
Licence MIT	http://www.opensource.org/licenses/mit-license.php
Licence Apache 2.0	http://www.opensource.org/licenses/apache2.0.php
Licence RPL	http://www.opensource.org/licenses/rpl.php
Licence OSL	http://www.opensource.org/licenses/osl-3.0.php
Histoire de l'informatique	http://histoire.info.online.fr/
Netcraft	http://news.netcraft.com/archives/2006/03/index.htm

Annexes

1 The Open Source Definition

Version 1.9

> *The indented, italicized sections below appear as annotations to the Open Source Definition (OSD) and are **not** a part of the OSD.*

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

> ***Rationale:** By constraining the license to require free redistribution, we eliminate the temptation to throw away many long-term gains in order to make a few short-term sales dollars. If we didn't do this, there would be lots of pressure for cooperators to defect.*

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost–preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

> ***Rationale:** We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.*

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

> ***Rationale:** The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.*

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

> ***Rationale:** Encouraging lots of improvement is a good thing, but users have a right to know who*

is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations.

*Accordingly, an open-source license **must** guarantee that source be readily available, but **may** require that it be distributed as pristine base sources plus patches. In this way, "unofficial" changes can be made available but readily distinguished from the base source.*

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

***Rationale:** In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process.*

Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

***Rationale:** The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.*

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

***Rationale:** This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.*

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

***Rationale:** This clause forecloses yet another class of license traps.*

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

***Rationale:** Distributors of open-source software have the right to make their own choices about their own software.*

Yes, the GPL is conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

> ***Rationale:*** *This provision is aimed specifically aimed at licenses which require an explicit gesture of assent in order to establish a contract between licensor and licensee. Provisions mandating so-called "click-wrap" may conflict with important methods of software distribution such as FTP download, CD-ROM anthologies, and web mirroring; such provisions may also hinder code re-use. Conformant licenses must allow for the possibility that (a) redistribution of the software will take place over non-Web channels that do not support click-wrapping of the download, and that (b) the covered code (or re-used portions of covered code) may run in a non-GUI environment that cannot support popup dialogues.*

2 Licences

2.1 The GNU General Public License (GPL)

Version 2, June 1991

Preamble
The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

> a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
>
> b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
>
> c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written

entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

> a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
>
> b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
>
> c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> One line to give the program's name and a brief idea of what it does.
> Copyright (C) <year> <name of author>
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> signature of Ty Coon, 1 April 1989
> Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

2.2 Open Software License ("OSL") v. 3.0

This Open Software License (the "License") applies to any original work of authorship (the "Original Work") whose owner (the "Licensor") has placed the following licensing notice adjacent to the copyright notice for the Original Work:

Licensed under the Open Software License version 3.0

1. **Grant of Copyright License**. Licensor grants You a worldwide, royalty-free, non-exclusive, sublicensable license, for the duration of the copyright, to do the following:
    a. to reproduce the Original Work in copies, either alone or as part of a collective work;
    b. to translate, adapt, alter, transform, modify, or arrange the Original Work, thereby creating derivative works ("Derivative Works") based upon the Original Work;
    c. to distribute or communicate copies of the Original Work and Derivative Works to the public, with the proviso that copies of Original Work or Derivative Works that You distribute or communicate shall be licensed under this Open Software License;
    d. to perform the Original Work publicly; and
    e. to display the Original Work publicly.
2. **Grant of Patent License**. Licensor grants You a worldwide, royalty-free, non-exclusive, sublicensable license, under patent claims owned or controlled by the Licensor that are embodied in the Original Work as furnished by the Licensor, for the duration of the patents, to make, use, sell, offer for sale, have made, and import the Original Work and Derivative Works.
3. **Grant of Source Code License**. The term "Source Code" means the preferred form of the Original Work for making modifications to it and all available documentation describing how to modify the Original Work. Licensor agrees to provide a machine-readable copy of the Source Code of the Original Work along with each copy of the Original Work that Licensor distributes. Licensor reserves the right to satisfy this obligation by placing a machine-readable copy of the Source Code in an information repository reasonably calculated to permit inexpensive and convenient access by You for as long as Licensor continues to distribute the Original Work.

4. **Exclusions From License Grant**. Neither the names of Licensor, nor the names of any contributors to the Original Work, nor any of their trademarks or service marks, may be used to endorse or promote products derived from this Original Work without express prior permission of the Licensor. Except as expressly stated herein, nothing in this License grants any license to Licensor's trademarks, copyrights, patents, trade secrets or any other intellectual property. No patent license is granted to make, use, sell, offer for sale, have made, or import embodiments of any patent claims other than the licensed claims defined in Section 2. No license is granted to the trademarks of Licensor even if such marks are included in the Original Work. Nothing in this License shall be interpreted to prohibit Licensor from licensing under terms different from this License any Original Work that Licensor otherwise would have a right to license.
5. **External Deployment**. The term "External Deployment" means the use, distribution, or communication of the Original Work or Derivative Works in any way such that the Original Work or Derivative Works may be used by anyone other than You, whether those works are distributed or communicated to those persons or made available as an application intended for use over a network. As an express condition for the grants of license hereunder, You must treat any External Deployment by You of the Original Work or a Derivative Work as a distribution under section 1(c).
6. **Attribution Rights**. You must retain, in the Source Code of any Derivative Works that You create, all copyright, patent, or trademark notices from the Source Code of the Original Work, as well as any notices of licensing and any descriptive text identified therein as an "Attribution Notice." You must cause the Source Code for any Derivative Works that You create to carry a prominent Attribution Notice reasonably calculated to inform recipients that You have modified the Original Work.
7. **Warranty of Provenance and Disclaimer of Warranty**. Licensor warrants that the copyright in and to the Original Work and the patent rights granted herein by Licensor are owned by the Licensor or are sublicensed to You under the terms of this License with the permission of the contributor(s) of those copyrights and patent rights. Except as expressly stated in the immediately preceding sentence, the Original Work is provided under this License on an "AS IS" BASIS and WITHOUT WARRANTY, either express or implied, including, without limitation, the warranties of non-infringement, merchantability or fitness for a particular purpose. THE ENTIRE RISK AS TO THE QUALITY OF THE ORIGINAL WORK IS WITH YOU. This DISCLAIMER OF WARRANTY constitutes an essential part of this License. No license to the Original Work is granted by this License except under this disclaimer.
8. **Limitation of Liability**. Under no circumstances and under no legal theory, whether in tort (including negligence), contract, or otherwise, shall the Licensor be liable to anyone for any indirect, special, incidental, or consequential damages of any character arising as a result of this License or the use of the Original Work including, without limitation, damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses. This limitation of liability shall not apply to the extent applicable law prohibits such limitation.
9. **Acceptance and Termination**. If, at any time, You expressly assented to this License, that assent indicates your clear and irrevocable acceptance of this License and all of its terms and conditions. If You distribute or communicate copies of the Original Work or

a Derivative Work, You must make a reasonable effort under the circumstances to obtain the express assent of recipients to the terms of this License. This License conditions your rights to undertake the activities listed in Section 1, including your right to create Derivative Works based upon the Original Work, and doing so without honoring these terms and conditions is prohibited by copyright law and international treaty. Nothing in this License is intended to affect copyright exceptions and limitations (including 'fair use' or 'fair dealing'). This License shall terminate immediately and You may no longer exercise any of the rights granted to You by this License upon your failure to honor the conditions in Section 1(c).

10. **Termination for Patent Action**. This License shall terminate automatically and You may no longer exercise any of the rights granted to You by this License as of the date You commence an action, including a cross-claim or counterclaim, against Licensor or any licensee alleging that the Original Work infringes a patent. This termination provision shall not apply for an action alleging patent infringement by combinations of the Original Work with other software or hardware.

11. **Jurisdiction, Venue and Governing Law**. Any action or suit relating to this License may be brought only in the courts of a jurisdiction wherein the Licensor resides or in which Licensor conducts its primary business, and under the laws of that jurisdiction excluding its conflict-of-law provisions. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any use of the Original Work outside the scope of this License or after its termination shall be subject to the requirements and penalties of copyright or patent law in the appropriate jurisdiction. This section shall survive the termination of this License.

12. **Attorneys Fees**. In any action to enforce the terms of this License or seeking damages relating thereto, the prevailing party shall be entitled to recover its costs and expenses, including, without limitation, reasonable attorneys' fees and costs incurred in connection with such action, including any appeal of such action. This section shall survive the termination of this License.

13. **Miscellaneous**. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable.

14. **Definition of "You" in This License**. "You" throughout this License, whether in upper or lower case, means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with you. For purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

15. **Right to Use.** You may use the Original Work in all ways not otherwise restricted or conditioned by this License or by law, and Licensor promises not to interfere with or be responsible for such uses by You.

16. **Modification of This License**. This License is Copyright Â© 2005 Lawrence Rosen. Permission is granted to copy, distribute, or communicate this License without modification. Nothing in this License permits You to modify this License as applied to the Original Work or to Derivative Works. However, You may modify the text of this

License and copy, distribute or communicate your modified version (the "Modified License") and apply it to other original works of authorship subject to the following conditions: (i) You may not indicate in any way that your Modified License is the "Open Software License" or "OSL" and you may not use those names in the name of your Modified License; (ii) You must replace the notice specified in the first paragraph above with the notice "Licensed under " or with a notice of your own that is not confusingly similar to the notice in this License; and (iii) You may not claim that your original works are open source software unless your Modified License has been approved by Open Source Initiative (OSI) and You comply with its license review and certification process.

2.3 Mozilla Public License 1.1 (MPL 1.1)

1. Definitions.

1.0.1. "Commercial Use" means distribution or otherwise making the Covered Code available to a third party.

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. "Contributor Version" means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. "Covered Code" means the Original Code or Modifications or the combination of the Original Code and Modifications, in each case including portions thereof**.**

1.4. "Electronic Distribution Mechanism" means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. "Executable" means Covered Code in any form other than Source Code.

1.6. "Initial Developer" means the individual or entity identified as the Initial Developer in the Source Code notice required by **Exhibit A**.

1.7. "Larger Work" means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. "License" means this document.

1.8.1. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. "Modifications" means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

> **A.** Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.
>
> **B.** Any new file that contains any part of the Original Code or previous Modifications.

1.10. "Original Code" means Source Code of computer software code which is described in the Source Code notice required by **Exhibit A** as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. "Patent Claims" means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. "Source Code" means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable,

or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor's choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. "You" (or "Your") means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, "You" includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. Source Code License.

2.1. The Initial Developer Grant.
The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under Patents Claims infringed by the making, using or selling of Original Code, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Code (or portions thereof).

(c) the licenses granted in this Section 2.1(a) and (b) are effective on the date Initial Developer first distributes Original Code under the terms of this License.

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: 1) for code that You delete from the Original Code; 2) separate from the Original Code; or 3) for infringements caused by: i) the modification of the Original Code or ii) the combination of the Original Code with other software or devices.

2.2. Contributor Grant.
Subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: 1) Modifications made by that Contributor (or portions thereof); and 2) the combination of

Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) the licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first makes Commercial Use of the Covered Code.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: 1) for any code that Contributor has deleted from the Contributor Version; 2) separate from the Contributor Version; 3) for infringements caused by: i) third party modifications of Contributor Version or ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or 4) under Patent Claims infringed by Covered Code in the absence of Modifications made by that Contributor.

3. Distribution Obligations.

3.1. Application of License.
The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section **2.2**. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section **6.1**, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section **3.5**.

3.2. Availability of Source Code.
Any Modification which You create or to which You contribute must be made available in Source Code form under the terms of this License either on the same media as an Executable version or via an accepted Electronic Distribution Mechanism to anyone to whom you made an Executable version available; and if made available via Electronic Distribution Mechanism, must remain available for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available to such recipients. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party.

3.3. Description of Modifications.
You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters

(a) Third Party Claims.
If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source

Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs.
If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

(c) Representations.
Contributor represents that, except as disclosed pursuant to Section 3.4(a) above, Contributor believes that Contributor's Modifications are Contributor's original creation(s) and/or Contributor has sufficient rights to grant the rights conveyed by this License.

3.5. Required Notices.
You must duplicate the notice in **Exhibit A** in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your name as a Contributor to the notice described in **Exhibit A**. You must also duplicate this License in any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear than any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.6. Distribution of Executable Versions.
You may distribute Covered Code in Executable form only if the requirements of Section **3.1-3.5** have been met for that Covered Code, and if You include a notice stating that the Source Code version of the Covered Code is available under the terms of this License, including a description of how and where You have fulfilled the obligations of Section **3.2**. The notice must be conspicuously included in any notice in an Executable version, related documentation or collateral in which You describe recipients' rights relating to the Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not

attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.7. Larger Works.
You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

4. Inability to Comply Due to Statute or Regulation.

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section **3.4** and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Application of this License.

This License applies to code to which the Initial Developer has attached the notice in **Exhibit A** and to related Covered Code.

6. Versions of the License.

6.1. New Versions.
Netscape Communications Corporation ("Netscape") may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions.
Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License.

6.3. Derivative Works.
If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrases "Mozilla", "MOZILLAPL", "MOZPL", "Netscape", "MPL", "NPL" or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the Mozilla Public License and Netscape Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in **Exhibit**

A shall not of themselves be deemed to be modifications of this License.)

7. DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8. TERMINATION.

8.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2. If You initiate litigation by asserting a patent infringement claim (excluding declatory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You file such action is referred to as "Participant") alleging that:

(a) such Participant's Contributor Version directly or indirectly infringes any patent, then any and all rights granted by such Participant to You under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively, unless if within 60 days after receipt of notice You either: (i) agree in writing to pay Participant a mutually agreeable reasonable royalty for Your past and future use of Modifications made by such Participant, or (ii) withdraw Your litigation claim with respect to the Contributor Version against such Participant. If within 60 days of notice, a reasonable royalty and payment arrangement are not mutually agreed upon in writing by the parties or the litigation claim is not withdrawn, the rights granted by Participant to You under Sections 2.1 and/or 2.2 automatically terminate at the expiration of the 60 day notice period specified above.

(b) any software, hardware, or device, other than such Participant's Contributor Version, directly or indirectly infringes any patent, then any rights granted to You by such Participant under Sections 2.1(b) and 2.2(b) are revoked effective as of the date You first made, used, sold, distributed, or had made, Modifications made by that Participant.

8.3. If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the

amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9. LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10. U.S. GOVERNMENT END USERS.

The Covered Code is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Code with only those rights set forth herein.

11. MISCELLANEOUS.

This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. With respect to disputes in which at least one party is a citizen of, or an entity chartered or registered to do business in the United States of America, any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California, with venue lying in Santa Clara County, California, with the losing party responsible for costs, including without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License.

12. RESPONSIBILITY FOR CLAIMS.

As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

13. MULTIPLE-LICENSED CODE.

Initial Developer may designate portions of the Covered Code as Multiple-Licensed. Multiple-Licensed means that the Initial Developer permits you to utilize portions of the Covered Code under Your choice of the MPL or the alternative licenses, if any, specified by the Initial Developer in the file described in Exhibit A.

EXHIBIT A -Mozilla Public License.

``The contents of this file are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at
http://www.mozilla.org/MPL/

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF
ANY KIND, either express or implied. See the License for the specific language governing rights and
limitations under the License.

The Original Code is _____.

The Initial Developer of the Original Code is _____. Portions created by
_____ are Copyright (C) _____ _____.
All Rights
Reserved.

Contributor(s): _____.

Alternatively, the contents of this file may be used under the terms of the _____ license (the [___] License), in which case the provisions of [_____] License are applicable instead of those above. If you wish to allow use of your version of this file only under the terms of the [____] License and not to allow others to use your version of this file under the MPL, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the [___] License. If you do not delete the provisions above, a recipient may use your version of this file under either the MPL or the [___] License."

[NOTE: The text of this Exhibit A may differ slightly from the text of the notices in the Source Code files of the Original Code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

2.4 The BSD License

The following is a BSD license template. To generate your own license, change the values of OWNER, ORGANIZATION and YEAR from their original values as given here, and substitute your own.

Note: The advertising clause in the license appearing on BSD Unix files was officially rescinded by the Director of the Office of Technology Licensing of the University of California on July 22 1999. He states that clause 3 is "hereby deleted in its entirety."

Note the new BSD license is thus equivalent to the MIT License, except for the no-endorsement final clause.

<OWNER> = Regents of the University of California
<ORGANIZATION> = University of California, Berkeley
<YEAR> = 1998

In the original BSD license, both occurrences of the phrase "COPYRIGHT HOLDERS AND CONTRIBUTORS" in the disclaimer read "REGENTS AND CONTRIBUTORS".

Here is the license template:

Copyright (c) <YEAR>, <OWNER>
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

2.5 The MIT License

Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

2.6 Apache License, Version 2.0

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

> To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

2.7 RECIPROCAL PUBLIC LICENSE

Version 1.1, November 1, 2002

PREAMBLE

This Preamble is intended to describe, in plain English, the nature, intent, and scope of this License. However, this Preamble is not a part of this License. The legal effect of this License is dependent only upon the terms of the License and not this Preamble.

This License is based on the concept of reciprocity. In exchange for being granted certain rights under the terms of this License to Licensor's Software, whose Source Code You have access to, You are required to reciprocate by providing equal access and rights to all third parties to the Source Code of any Modifications, Derivative Works, and Required Components for execution of same (collectively defined as Extensions) that You Deploy by Deploying Your Extensions under the terms of this License. In this fashion the available Source Code related to the original Licensed Software is enlarged for the benefit of everyone.

Under the terms of this License You may:

a. Distribute the Licensed Software exactly as You received it under the terms of this License either alone or as a component of an aggregate software distribution containing programs from several different sources without payment of a royalty or other fee.

b. Use the Licensed Software for any purpose consistent with the rights granted by this License, but the Licensor is not providing You any warranty whatsoever, nor is the Licensor accepting any liability in the event that the Licensed Software doesn't work properly or causes You any injury or damages.

c. Create Extensions to the Licensed Software consistent with the rights granted by this License, provided that You make the Source Code to any Extensions You Deploy available to all third parties under the terms of this License, document Your Modifications clearly, and title all Extensions distinctly from the Licensed Software.

d. Charge a fee for warranty or support, or for accepting indemnity or liability obligations for Your customers.

Under the terms of this License You may not:

a. Charge for the Source Code to the Licensed Software, or Your Extensions, other than a nominal fee not to exceed Your cost for reproduction and distribution where such reproduction and distribution involve physical media.

b. Modify or delete any pre-existing copyright notices, change notices, or License text in the Licensed Software.

c. Assert any patent claims against the Licensor or Contributors, or which would in any way restrict the ability of any third party to use the Licensed Software or portions thereof in any form under the terms of this License, or Your rights to the Licensed Software under this

License automatically terminate.

d. Represent either expressly or by implication, appearance, or otherwise that You represent Licensor or Contributors in any capacity or that You have any form of legal association by virtue of this License.

Under the terms of this License You must:

a. Document any Modifications You make to the Licensed Software including the nature of the change, the authors of the change, and the date of the change. This documentation must appear both in the Source Code and in a text file titled "CHANGES" distributed with the Licensed Software and Your Extensions.

b. Make the Source Code for any Extensions You Deploy available in a timely fashion via an Electronic Distribution Mechanism such as FTP or HTTP download.

c. Notify the Licensor of the availability of Source Code to Your Extensions in a timely fashion and include in such notice a brief description of the Extensions, the distinctive title used, and instructions on how to acquire the Source Code and future updates.

d. Grant Licensor and all third parties a world-wide, non-exclusive, royalty-free license under any intellectual property rights owned or controlled by You to use, reproduce, display, perform, modify, sublicense, and distribute Your Extensions, in any form, under the terms of this License.

LICENSE TERMS

1.0 General; Applicability & Definitions. This Reciprocal Public License Version 1.1 ("License") applies to any programs or other works as well as any and all updates or maintenance releases of said programs or works ("Software") not already covered by this License which the Software copyright holder ("Licensor") makes publicly available containing a Notice (hereinafter defined) from the Licensor specifying or allowing use or distribution under the terms of this License. As used in this License and Preamble:

1.1 "Contributor" means any person or entity who created or contributed to the creation of an Extension.

1.2 "Deploy" means to use, Serve, sublicense or distribute Licensed Software other than for Your internal Research and/or Personal Use, and includes without limitation, any and all internal use or distribution of Licensed Software within Your business or organization other than for Research and/or Personal Use, as well as direct or indirect sublicensing or distribution of Licensed Software by You to any third party in any form or manner.

1.3 "Derivative Works" as used in this License is defined under U.S. copyright law.

1.4 "Electronic Distribution Mechanism" means a mechanism generally accepted in the software development community for the electronic transfer of data such as download from an FTP or web site, where such mechanism is publicly accessible.

1.5 "Extensions" means any Modifications, Derivative Works, or Required Components as those terms are defined in this License.

1.6 "License" means this Reciprocal Public License.

1.7 "Licensed Software" means any Software licensed pursuant to this License. Licensed Software also includes all previous Extensions from any Contributor that You receive.

1.8 "Licensor" means the copyright holder of any Software previously uncovered by this License who releases the Software under the terms of this License.

1.9 "Modifications" means any additions to or deletions from the substance or structure of (i) a file or other storage containing Licensed Software, or (ii) any new file or storage that contains any part of Licensed Software, or (iii) any file or storage which replaces or otherwise alters the original functionality of Licensed Software at runtime.

1.10 "Notice" means the notice contained in EXHIBIT A.

1.11 "Personal Use" means use of Licensed Software by an individual solely for his or her personal, private and non-commercial purposes. An individual's use of Licensed Software in his or her capacity as an officer, employee, member, independent contractor or agent of a corporation, business or organization (commercial or non-commercial) does not qualify as Personal Use.

1.12 "Required Components" means any text, programs, scripts, schema, interface definitions, control files, or other works created by You which are required by a third party of average skill to successfully install and run Licensed Software containing Your Modifications, or to install and run Your Derivative Works.

1.13 "Research" means investigation or experimentation for the purpose of understanding the nature and limits of the Licensed Software and its potential uses.

1.14 "Serve" means to deliver Licensed Software and/or Your Extensions by means of a computer network to one or more computers for purposes of execution of Licensed Software and/or Your Extensions.

1.15 "Software" means any computer programs or other works as well as any updates or maintenance releases of those programs or works which are distributed publicly by Licensor.

1.16 "Source Code" means the preferred form for making modifications to the Licensed Software and/or Your Extensions, including all modules contained therein, plus any associated text, interface definition files, scripts used to control compilation and installation of an executable program or other components required by a third party of average skill to build a running version of the Licensed Software or Your Extensions.

1.17 "You" or "Your" means an individual or a legal entity exercising rights under this License. For legal entities, "You" or "Your" includes any entity which controls, is controlled by, or is under common control with, You, where "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of fifty percent (50%) or more of the outstanding shares or beneficial ownership of such entity.

2.0 Acceptance Of License. You are not required to accept this License since you have not signed it, however nothing else grants you permission to use, copy, distribute, modify, or create derivatives of either the Software or any Extensions created by a Contributor. These actions are prohibited by law if you do not accept this License. Therefore, by performing any of these actions You indicate Your acceptance of this License and Your agreement to be bound by all its terms and conditions. IF YOU DO NOT AGREE WITH ALL THE TERMS AND CONDITIONS OF THIS LICENSE DO NOT USE, MODIFY, CREATE DERIVATIVES, OR DISTRIBUTE THE SOFTWARE. IF IT IS IMPOSSIBLE FOR YOU TO COMPLY WITH ALL THE TERMS AND CONDITIONS OF THIS LICENSE THEN YOU CAN NOT USE, MODIFY, CREATE DERIVATIVES, OR DISTRIBUTE THE

SOFTWARE.

3.0 Grant of License From Licensor. Subject to the terms and conditions of this License, Licensor hereby grants You a world-wide, royalty-free, non-exclusive license, subject to Licensor's intellectual property rights, and any third party intellectual property claims derived from the Licensed Software under this License, to do the following:

3.1 Use, reproduce, modify, display, perform, sublicense and distribute Licensed Software and Your Extensions in both Source Code form or as an executable program.

3.2 Create Derivative Works (as that term is defined under U.S. copyright law) of Licensed Software by adding to or deleting from the substance or structure of said Licensed Software.

3.3 Under claims of patents now or hereafter owned or controlled by Licensor, to make, use, have made, and/or otherwise dispose of Licensed Software or portions thereof, but solely to the extent that any such claim is necessary to enable You to make, use, have made, and/or otherwise dispose of Licensed Software or portions thereof.

3.4 Licensor reserves the right to release new versions of the Software with different features, specifications, capabilities, functions, licensing terms, general availability or other characteristics. Title, ownership rights, and intellectual property rights in and to the Licensed Software shall remain in Licensor and/or its Contributors.

4.0 Grant of License From Contributor. By application of the provisions in Section 6 below, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license, subject to said Contributor's intellectual property rights, and any third party intellectual property claims derived from the Licensed Software under this License, to do the following:

4.1 Use, reproduce, modify, display, perform, sublicense and distribute any Extensions Deployed by such Contributor or portions thereof, in both Source Code form or as an executable program, either on an unmodified basis or as part of Derivative Works.

4.2 Under claims of patents now or hereafter owned or controlled by Contributor, to make, use, have made, and/or otherwise dispose of Extensions or portions thereof, but solely to the extent that any such claim is necessary to enable You to make, use, have made, and/or otherwise dispose of Contributor's Extensions or portions thereof.

5.0 Exclusions From License Grant. Nothing in this License shall be deemed to grant any rights to trademarks, copyrights, patents, trade secrets or any other intellectual property of Licensor or any Contributor except as expressly stated herein. Except as expressly stated in Sections 3 and 4, no other patent rights, express or implied, are granted herein. Your Extensions may require additional patent licenses from Licensor or Contributors which each may grant in its sole discretion. No right is granted to the trademarks of Licensor or any Contributor even if such marks are included in the Licensed Software. Nothing in this License shall be interpreted to prohibit Licensor from licensing under different terms from this License any code that Licensor otherwise would have a right to license.

5.1 You expressly acknowledge and agree that although Licensor and each Contributor grants the licenses to their respective portions of the Licensed Software set forth herein, no assurances are provided by Licensor or any Contributor that the Licensed Software does not infringe the patent or other intellectual property rights of any other entity. Licensor and each Contributor disclaim any liability to You for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, You hereby assume sole responsibility to secure any

other intellectual property rights needed, if any. For example, if a third party patent license is required to allow You to distribute the Licensed Software, it is Your responsibility to acquire that license before distributing the Licensed Software.

6.0 Your Obligations And Grants. In consideration of, and as an express condition to, the licenses granted to You under this License You hereby agree that any Modifications, Derivative Works, or Required Components (collectively Extensions) that You create or to which You contribute are governed by the terms of this License including, without limitation, Section 4. Any Extensions that You create or to which You contribute must be Deployed under the terms of this License or a future version of this License released under Section 7. You hereby grant to Licensor and all third parties a world-wide, non-exclusive, royalty-free license under those intellectual property rights You own or control to use, reproduce, display, perform, modify, create derivatives, sublicense, and distribute Your Extensions, in any form. Any Extensions You make and Deploy must have a distinct title so as to readily tell any subsequent user or Contributor that the Extensions are by You. You must include a copy of this License with every copy of the Extensions You distribute. You agree not to offer or impose any terms on any Source Code or executable version of the Licensed Software, or its Extensions that alter or restrict the applicable version of this License or the recipients' rights hereunder.

6.1 Availability of Source Code. You must make available, under the terms of this License, the Source Code of the Licensed Software and any Extensions that You Deploy, either on the same media as You distribute any executable or other form of the Licensed Software, or via an Electronic Distribution Mechanism. The Source Code for any version of Licensed Software, or its Extensions that You Deploy must be made available at the time of Deployment and must remain available for as long as You Deploy the Extensions or at least twelve (12) months after the date You Deploy, whichever is longer. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party. You may not charge a fee for the Source Code distributed under this Section in excess of Your actual cost of duplication and distribution where such duplication and distribution involve physical media.

6.2 Description of Modifications. You must cause any Modifications that You create or to which You contribute, to update the file titled "CHANGES" distributed with Licensed Software documenting the additions, changes or deletions You made, the authors of such Modifications, and the dates of any such additions, changes or deletions. You must also cause a cross-reference to appear in the Source Code at the location of each change. You must include a prominent statement that the Modifications are derived, directly or indirectly, from the Licensed Software and include the names of the Licensor and any Contributor to the Licensed Software in (i) the Source Code and (ii) in any notice displayed by the Licensed Software You distribute or in related documentation in which You describe the origin or ownership of the Licensed Software. You may not modify or delete any pre-existing copyright notices, change notices or License text in the Licensed Software.

6.3 Intellectual Property Matters.

a. Third Party Claims. If You have knowledge that a license to a third party's intellectual property right is required to exercise the rights granted by this License, You must include a text file with the Source Code distribution titled "LEGAL" that describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If You obtain such knowledge after You make any Extensions available as described in Section 6.1,

You shall promptly modify the LEGAL file in all copies You make available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Licensed Software from You that new knowledge has been obtained.

b. Contributor APIs. If Your Extensions include an application programming interface ("API") and You have knowledge of patent licenses that are reasonably necessary to implement that API, You must also include this information in the LEGAL file.

c. Representations. You represent that, except as disclosed pursuant to 6.3(a) above, You believe that any Extensions You distribute are Your original creations and that You have sufficient rights to grant the rights conveyed by this License.

6.4 Required Notices.

a. License Text. You must duplicate this License in any documentation You provide along with the Source Code of any Extensions You create or to which You contribute, wherever You describe recipients' rights relating to Licensed Software. You must duplicate the notice contained in EXHIBIT A (the "Notice") in each file of the Source Code of any copy You distribute of the Licensed Software and Your Extensions. If You create an Extension, You may add Your name as a Contributor to the text file titled "CONTRIB" distributed with the Licensed Software along with a description of the contribution. If it is not possible to put the Notice in a particular Source Code file due to its structure, then You must include such Notice in a location (such as a relevant directory file) where a user would be likely to look for such a notice.

b. Source Code Availability. You must notify Licensor within one (1) month of the date You initially Deploy of the availability of Source Code to Your Extensions and include in such notification the name under which you Deployed Your Extensions, a description of the Extensions, and instructions on how to acquire the Source Code, including instructions on how to acquire updates over time. Should such instructions change you must provide Licensor with revised instructions within one (1) month of the date of change. Should you be unable to notify Licensor directly, you must provide notification by posting to appropriate news groups, mailing lists, or web sites where a search engine would reasonably be expected to index them.

6.5 Additional Terms. You may choose to offer, and charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Licensed Software. However, You may do so only on Your own behalf, and not on behalf of the Licensor or any Contributor. You must make it clear that any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Licensor and every Contributor for any liability plus attorney fees, costs, and related expenses due to any such action or claim incurred by the Licensor or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

6.6 Conflicts With Other Licenses. Where any portion of Your Extensions, by virtue of being Derivative Works of another product or similar circumstance, fall under the terms of another license, the terms of that license should be honored however You must also make Your Extensions available under this License. If the terms of this License continue to conflict with the terms of the other license you may write the Licensor for permission to resolve the conflict in a fashion that remains consistent with the intent of this License. Such permission will be granted at the sole discretion of the Licensor.

7.0 Versions of This License. Licensor may publish from time to time revised and/or new

versions of the License. Once Licensed Software has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Licensed Software under the terms of any subsequent version of the License published by Licensor. No one other than Licensor has the right to modify the terms applicable to Licensed Software created under this License.

7.1 If You create or use a modified version of this License, which You may do only in order to apply it to software that is not already Licensed Software under this License, You must rename Your license so that it is not confusingly similar to this License, and must make it clear that Your license contains terms that differ from this License. In so naming Your license, You may not use any trademark of Licensor or of any Contributor. Should Your modifications to this License be limited to alteration of EXHIBIT A purely for purposes of adjusting the Notice You require of licensees, You may continue to refer to Your License as the Reciprocal Public License or simply the RPL.

8.0 Disclaimer of Warranty. LICENSED SOFTWARE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE LICENSED SOFTWARE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. FURTHER THERE IS NO WARRANTY MADE AND ALL IMPLIED WARRANTIES ARE DISCLAIMED THAT THE LICENSED SOFTWARE MEETS OR COMPLIES WITH ANY DESCRIPTION OF PERFORMANCE OR OPERATION, SAID COMPATIBILITY AND SUITABILITY BEING YOUR RESPONSIBILITY. LICENSOR DISCLAIMS ANY WARRANTY, IMPLIED OR EXPRESSED, THAT ANY CONTRIBUTOR'S EXTENSIONS MEET ANY STANDARD OF COMPATIBILITY OR DESCRIPTION OF PERFORMANCE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LICENSED SOFTWARE IS WITH YOU. SHOULD LICENSED SOFTWARE PROVE DEFECTIVE IN ANY RESPECT, YOU (AND NOT THE LICENSOR OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. UNDER THE TERMS OF THIS LICENSOR WILL NOT SUPPORT THIS SOFTWARE AND IS UNDER NO OBLIGATION TO ISSUE UPDATES TO THIS SOFTWARE. LICENSOR HAS NO KNOWLEDGE OF ERRANT CODE OR VIRUS IN THIS SOFTWARE, BUT DOES NOT WARRANT THAT THE SOFTWARE IS FREE FROM SUCH ERRORS OR VIRUSES. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF LICENSED SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

9.0 Limitation of Liability. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL THE LICENSOR, ANY CONTRIBUTOR, OR ANY DISTRIBUTOR OF LICENSED SOFTWARE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE

EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10.0 High Risk Activities. THE LICENSED SOFTWARE IS NOT FAULT-TOLERANT AND IS NOT DESIGNED, MANUFACTURED, OR INTENDED FOR USE OR DISTRIBUTION AS ON-LINE CONTROL EQUIPMENT IN HAZARDOUS ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS IN THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR COMMUNICATIONS SYSTEMS, AIR TRAFFIC CONTROL, DIRECT LIFE SUPPORT MACHINES, OR WEAPONS SYSTEMS, IN WHICH THE FAILURE OF THE LICENSED SOFTWARE COULD LEAD DIRECTLY TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE ("HIGH RISK ACTIVITIES"). LICENSOR AND CONTRIBUTORS SPECIFICALLY DISCLAIM ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS FOR HIGH RISK ACTIVITIES.

11.0 Responsibility for Claims. As between Licensor and Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License which specifically disclaims warranties and limits any liability of the Licensor. This paragraph is to be used in conjunction with and controlled by the Disclaimer Of Warranties of Section 8, the Limitation Of Damages in Section 9, and the disclaimer against use for High Risk Activities in Section 10. The Licensor has thereby disclaimed all warranties and limited any damages that it is or may be liable for. You agree to work with Licensor and Contributors to distribute such responsibility on an equitable basis consistent with the terms of this License including Sections 8, 9, and 10. Nothing herein is intended or shall be deemed to constitute any admission of liability.

12.0 Termination. This License and all rights granted hereunder will terminate immediately in the event of the circumstances described in Section 13.6 or if applicable law prohibits or restricts You from fully and or specifically complying with Sections 3, 4 and/or 6, or prevents the enforceability of any of those Sections, and You must immediately discontinue any use of Licensed Software.

12.1 Automatic Termination Upon Breach. This License and the rights granted hereunder will terminate automatically if You fail to comply with the terms herein and fail to cure such breach within thirty (30) days of becoming aware of the breach. All sublicenses to the Licensed Software that are properly granted shall survive any termination of this License. Provisions that, by their nature, must remain in effect beyond the termination of this License, shall survive.

12.2 Termination Upon Assertion of Patent Infringement. If You initiate litigation by asserting a patent infringement claim (excluding declaratory judgment actions) against Licensor or a Contributor (Licensor or Contributor against whom You file such an action is referred to herein as "Respondent") alleging that Licensed Software directly or indirectly infringes any patent, then any and all rights granted by such Respondent to You under Sections 3 or 4 of this License shall terminate prospectively upon sixty (60) days notice from Respondent (the "Notice Period") unless within that Notice Period You either agree in writing (i) to pay Respondent a mutually agreeable reasonably royalty for Your past or future use of Licensed Software made by such Respondent, or (ii) withdraw Your litigation claim with respect to Licensed Software against such Respondent. If within said Notice Period a

reasonable royalty and payment arrangement are not mutually agreed upon in writing by the parties or the litigation claim is not withdrawn, the rights granted by Licensor to You under Sections 3 and 4 automatically terminate at the expiration of said Notice Period.

12.3 Reasonable Value of This License. If You assert a patent infringement claim against Respondent alleging that Licensed Software directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by said Respondent under Sections 3 and 4 shall be taken into account in determining the amount or value of any payment or license.

12.4 No Retroactive Effect of Termination. In the event of termination under this Section all end user license agreements (excluding licenses to distributors and resellers) that have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

13.0 Miscellaneous.

13.1 U.S. Government End Users. The Licensed Software is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Licensed Software with only those rights set forth herein.

13.2 Relationship of Parties. This License will not be construed as creating an agency, partnership, joint venture, or any other form of legal association between or among You, Licensor, or any Contributor, and You will not represent to the contrary, whether expressly, by implication, appearance, or otherwise.

13.3 Independent Development. Nothing in this License will impair Licensor's right to acquire, license, develop, subcontract, market, or distribute technology or products that perform the same or similar functions as, or otherwise compete with, Extensions that You may develop, produce, market, or distribute.

13.4 Consent To Breach Not Waiver. Failure by Licensor or Contributor to enforce any provision of this License will not be deemed a waiver of future enforcement of that or any other provision.

13.5 Severability. This License represents the complete agreement concerning the subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable.

13.6 Inability to Comply Due to Statute or Regulation. If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Licensed Software due to statute, judicial order, or regulation, then You cannot use, modify, or distribute the software.

13.7 Export Restrictions. You may be restricted with respect to downloading or otherwise acquiring, exporting, or reexporting the Licensed Software or any underlying information or technology by United States and other applicable laws and regulations. By downloading or by otherwise obtaining the Licensed Software, You are agreeing to be responsible for compliance with all applicable laws and regulations.

13.8 Arbitration, Jurisdiction & Venue. This License shall be governed by Colorado law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. You expressly agree that any dispute relating to this License shall be submitted to binding arbitration under the rules then prevailing of the American Arbitration Association. You further agree that Adams County, Colorado USA is proper venue and grant such arbitration proceeding jurisdiction as may be appropriate for purposes of resolving any dispute under this License. Judgement upon any award made in arbitration may be entered and enforced in any court of competent jurisdiction. The arbitrator shall award attorney's fees and costs of arbitration to the prevailing party. Should either party find it necessary to enforce its arbitration award or seek specific performance of such award in a civil court of competent jurisdiction, the prevailing party shall be entitled to reasonable attorney's fees and costs. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. You and Licensor expressly waive any rights to a jury trial in any litigation concerning Licensed Software or this License. Any law or regulation that provides that the language of a contract shall be construed against the drafter shall not apply to this License.

13.9 Entire Agreement. This License constitutes the entire agreement between the parties with respect to the subject matter hereof.

EXHIBIT A

The Notice below must appear in each file of the Source Code of any copy You distribute of the Licensed Software or any Extensions thereto, except as may be modified as allowed under the terms of Section 7.1

> Copyright (C) 1999-2002 Technical Pursuit Inc., All Rights Reserved. Patent Pending, Technical Pursuit Inc.
>
> Unless explicitly acquired and licensed from Licensor under the Technical Pursuit License ("TPL") Version 1.0 or greater, the contents of this file are subject to the Reciprocal Public License ("RPL") Version 1.1, or subsequent versions as allowed by the RPL, and You may not copy or use this file in either source code or executable form, except in compliance with the terms and conditions of the RPL.
>
> You may obtain a copy of both the TPL and the RPL (the "Licenses") from Technical Pursuit Inc. at http://www.technicalpursuit.com.
>
> All software distributed under the Licenses is provided strictly on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, AND TECHNICAL PURSUIT INC. HEREBY DISCLAIMS ALL SUCH WARRANTIES, INCLUDING WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, QUIET ENJOYMENT, OR NON-INFRINGEMENT. See the Licenses for specific language governing rights and limitations under the Licenses.

3 Questionnaire distribué au personnel d'EASI-WAL

Questionnaire Logiciels EASI-WAL

Nom	
Prénom	
Fonction	

Ce questionnaire a pour objectif de récolter vos impressions sur la suite logicielle utilisée à EASI-WAL. Grâce à vos réponses, il me sera possible de proposer des améliorations et de rendre compte des difficultés rencontrées lors d'une migration.

Firefox

Que pensez-vous du logiciel Firefox par rapport au logiciel utilisé anciennement, probablement Internet Explorer? Prise en main, utilisation courante, etc.

Thunderbird

Que pensez-vous du logiciel Thunderbird par rapport au logiciel utilisé anciennement, probablement Outlook ou Outlook Express? Prise en main, utilisation courante, etc.

OpenOffice

Que pensez-vous, de manière générale, d' OpenOffice?

Y a-t-il des actions que vous saviez faire avant que vous ne parvenez plus à réaliser?

La migration des documents, Microsoft Office vers OpenOffice, c'est-elle bien déroulée de votre point de vue?

Questionnaire distribué au personnel d'EASI-WAL

Le manque d'une formation se fait-elle ressentir sur votre travail quotidien?

Avez-vous rencontré d'autres difficultés à l'utilisation?

Phprojekt

Que pensez-vous de ce logiciel par rapport à ce que vous utilisiez anciennement?

Son utilisation est-elle viable pour vous à long terme? Et pourquoi?

Autres Remarques

4 Processus de Migration du guide IDA, [Choppy, 2003]

Idéalement, le processus de migration doit consister en les parties qui suivent. Certaines d'entre elles peuvent être menées en parallèle, notamment 2, 3 et 4.

1. monter une équipe avec les compétences nécessaires et un appui managérial. Cet appui est important pour éviter les résistances au changement depuis la norme des systèmes propriétaires. Il devra être suffisant pour permettre au moins la mise en place de pilotes représentatifs, ainsi un cas concret de base sera-t-il sans doute nécessaire avec peut-être un autre plus détaillé par la suite lorsque plus d'informations seront disponibles.
2. comprendre l'environnement cible, aussi bien le logiciel OSS[58] que l'architecture de base (cf. chapitre 8[59]), ainsi que les options et choix possibles. Cela implique la formation de l'équipe existante, du recrutement ou le recours à des consultants. Cela entraîne un certain coût initial et nécessite donc un appui managérial suffisant. Parfois l'on s'attend à ce qu'un logiciel gratuit puisse être compris et utilisé à coût zéro. **Ce n'est pas le cas**.
3. la migration est une opportunité pour re-visiter l'architecture de base aussi bien que les logiciels applicatifs. L'architecture préconisée au chapitre 24[58] s'appuie sur un contrôle centralisé et dispose d'un certain nombre d'avantages détaillés dans ledit chapitre. Il peut y avoir des implications financières de ces changements qui doivent être prises en compte.
4. il est très important que l'OSS soit compris. Un certain nombre de points doivent être totalement maîtrisés avant toute prise de décision :
   A. les implications des licences OSS doivent être clairement perçues en particulier si l'administration peut être amenée à distribuer des modifications aux logiciels. Se reporter aux documents indiqués dans l'introduction[166] pour les détails.
   B. lorsqu'il existe plusieurs choix pour une seule fonction - il existe par exemple au moins trois tableurs de qualité en OSS - les administrateurs doivent peser le pour et le contre de chaque produit.
   C. il faut étudier les différences entre les distributions. Certaines sont appuyées par des entreprises commerciales qui fournissent un support et des mises à jour. Certaines ont des caractéristiques spécifiques (par exemple, Gentoo propose une distribution fondée sur le code source qui permet à l'administration d'adapter aisément le logiciel pour le faire coller au mieux à ses besoins spécifiques). Toutes ces différences doivent être étudiées avant qu'un choix soit effectué.
   D. les administrateurs doivent déterminer le niveau de support nécessaire. Un support payant peut être obtenu dans certains cas auprès des développeurs de l'application ou de la distribution. Dans le cas contraire, des entreprises tierces peuvent fournir un support en raison de la disponibilité du code source et de nombreuses entreprises proposent un tel support. Il y a une différence marquante par rapport au marché du

58 Dans ce document, OSS signifie Logiciel Open Source et définit tant les logiciels open source que les logiciels libres.

59 Référence à un chapitre ou élément du Guide IDA de migration Open Source.

logiciel propriétaire où un support détaillé ne peut être fourni que par les entreprises qui ont le privilège d'avoir accès aux sources. Cela devient important si l'éditeur propriétaire cesse son activité sans fournir le code source. Si tout le reste échoue, de nombreuses applications disposent de listes de diffusion actives sur lesquelles les questions et demandes d'assistances reçoivent des réponses de personnes qui s'intéressent à celles-ci. La présence d'une liste de diffusion active et d'une communauté d'utilisateurs est souvent l'un des critères initiaux de sélection de composants logiciels.

5. analyser les systèmes existants. Cette donnée ne sera pas utile seulement pour réaliser la migration proprement dite mais elle sera aussi nécessaire pour bâtir un modèle de coût total de possession (TCO - Total cost of ownership) pour un cas concret détaillé.

   Il faut compiler l'inventaire de :

   A. pour chaque application utilisée :

      a) nom de l'application, numéro de version et point de contact,

      b) nombre d'utilisateurs,

      c) système d'exploitation, liste des systèmes d'exploitations possibles y compris les environnements de type Citrix,

      d) interdépendance avec d'autres applications aussi bien sur le client que sur le serveur,

      e) matériel nécessaire, en particulier matériels non standard ou spécifiques,

      f) protocoles de communications utilisés,

      g) formats de fichiers nécessaires,

      h) internationalisation et localisation nécessaires (il peut être nécessaire de gérer plusieurs langues et devises par exemple) ;

   B. contraintes sur les données (à interpréter au sens large y compris, par exemple, textes et tableaux, données sonores/vocales et visuelles ainsi que les bases de données) ; plus généralement, tout ce qui est destiné à être traité ou stocké par un ordinateur :

      a) contraintes d'interfaçage avec des systèmes ou utilisateurs hors du contrôle de l'administration,

      b) contraintes d'exploitabilité future des données, existence d'un répertoire des données de référence à accepter, nécessité d'applications spécifiques pour l'exploitation de celui-ci... Les données peuvent être divisées comme suit :

         i. données jetables - les jeter,

         ii. données à conserver existant dans un format ouvert tel que PDF ou PostScript ou pouvant aisément être transcrites dans l'un de ces formats. Le coût de transcription doit être évalué,

         iii. données à conserver existant dans un format propriétaire difficile à transcrire en format ouvert. Ces données peuvent nécessiter le maintien d'applications propriétaires spécifiques ; le coût de celles-ci doit être évalué, ainsi que le nombre de copies nécessaires sur la base de la

fréquence d'utilisation (par exemple, si l'information est rarement utilisée, une seule copie sur une machine centralisée peut suffire). Il peut également être nécessaire de maintenir un matériel spécifique pour utiliser ces applications ;

C. contraintes de sécurité

a) quel est le système actuel d'assignation des noms d'utilisateurs et mots de passe? Existe-t-il une structuration des noms d'utilisateurs? Laquelle? Quelle est la politique d'expiration des mots de passe?

b) certains systèmes nécessitent-ils une authentification supérieure à un couple utilisateur/mot de passe?

c) Quelles sont les règles administratives et gouvernementales d'utilisation des ordinateurs (par exemple, existe-t-il des restrictions quant à l'utilisation d'Internet et du courriel)?

d) Y a-t-il des dispositions sécuritaires qui nécessitent l'utilisation de matériel ou de logiciel spécifique?

6. Créer un cas concret de migration détaillé ; celui-ci sera fondé sur les données rassemblées ci-dessus et comportera notamment :

   A. le TCO de l'environnement existant pour une période de temps raisonnable (5 ans par exemple),

   B. le TCO d'environnements différents et le coût de migration vers chacun pour la même période,

   C. les forces et faiblesses de l'environnement existant et de chaque alternative (la feuille de calcul associée peut aider à effectuer la comparaison) ;

7. consulter les utilisateurs. Leur expliquer le raisonnement sous-jacent à la migration et les effets qui leur seront perceptibles. Prendre sérieusement en compte leurs besoins et leur permettre de « jouer » avec les nouveautés aussi tôt que possible. Plus tôt les utilisateurs sont impliqués, mieux la migration est acceptée. Il peut y avoir des nécessités légales dans certains pays mais cela doit être fait dans tous les cas pour faciliter l'introduction de ce qui peut constituer une modification significative dans le travail quotidien.

   Créer un pôle d'assistance qui puisse répondre aux besoins des utilisateurs. Plus tard, lorsque la migration est en cours, celui-ci pourra résoudre les problèmes et devenir un centre d'excellence et de bonnes pratiques. Créer un site intranet avec une section « astuces et guides » qui puisse être mise à jour par les utilisateurs eux-mêmes. Il est important que les utilisateurs se sentent impliqués et le site donnera aussi au pôle d'assistance une idée du genre de problèmes auxquels sont confrontés les utilisateurs.

8. en supposant réalisé le cas concret, commencer avec des projets pilotes à petite échelle, de préférence dans un environnement isolé avec un petit nombre d'utilisateurs. Ceux-ci fourniront notamment

   A. des données pour affiner les modèles de TCO,

   B. des réactions d'utilisateurs qui permettront de faciliter l'introduction d'autres systèmes,

C. la validation ou la modification de l'architecture cible et du cas concret ;

9. décider de la vitesse du processus de migration ; il est vraisemblable que les deux systèmes doivent cohabiter côte à côte durant un certain temps. Il est important d'avoir une stratégie de transition permettant à ceux-ci de fonctionner simultanément afin que les activités de production puissent continuer correctement durant la période de migration. Le remplacement de la dernière machine peut prendre beaucoup de temps (ou ne jamais arriver) donc la faisabilité de la coexistence peut être très importante. Les principales options sont :

   A. **Big Bang** : tous les utilisateurs passent au nouveau système le même jour. Dans la pratique, il est vraisemblable que la migration soit planifiée sur une fin de semaine ou durant les vacances. L'avantage réside dans l'absence de structures de double accès et de double maintenance par les équipes techniques. Les inconvénients résident dans un niveau de risque très élevé et dans la très forte mobilisation de ressources durant la migration. Ce schéma semble ne pouvoir être appliqué que dans les petites administrations. Dans la mesure du possible, **ÉVITEZ LA MIGRATION BIG BANG**. Ce type de migrations nécessite le contrôle de tant de variables qu'elles échouent en général. Si c'est le cas, ce n'est pas, en général, du fait d'une défaillance de l'OSS mais de celle du management.

   B. **Transition progressive par groupes** : les utilisateurs migrent par groupes. On fait migrer en général des groupes fonctionnels complets afin de minimiser les partages de données et les difficultés de travail de groupe. Les risques peuvent être limités et les ressources gérées par l'adéquation de la taille des groupes. Il peut être possible de réaliser un remplacement du matériel par roulement dans le même temps, en effectuant une mise à jour des postes de travail retirés d'un groupe avant de les attribuer au groupe suivant.

   C. **Transition individuelle** : similaire à la transition progressive par groupes mais avec une taille de groupe d'une personne. Cette méthode au goutte-à-goutte mobilise peu de ressources mais elle est peu efficace et sans doute peu adaptée aux grandes administrations. Elle peut cependant être appropriée aux projet pilotes.

10. Réaliser la migration de l'ensemble de l'administration. Cela implique une formation ultérieure des utilisateurs et de l'équipe technique.

11. Suivre les retours des utilisateurs et résoudre tous les problèmes qui surviennent. Certains besoins utilisateurs peuvent être suffisamment obscurs pour n'avoir pas été prévus à l'avance ni découvert durant les phases pilotes. S'assurer que des ressources suffisantes restent disponibles pour répondre à ce type de besoins après la transition. À tout instant, il peut être découvert une impossibilité de migration. Cela peut arriver par exemple en raison de l'existence d'applications critiques qui ne fonctionnent pas de manière satisfaisante dans un environnement OSS et dont le coût de réécriture serait trop élevé.

À tout instant, il peut être découvert une impossibilité de migration. Cela peut arriver par exemple en raison de l'existence d'applications critiques qui ne fonctionnent pas de manière satisfaisante dans un environnement OSS et dont le coût de réécriture serait trop élevé.